ECE 2300
Digital Logic & Computer Organization

Fall 2016

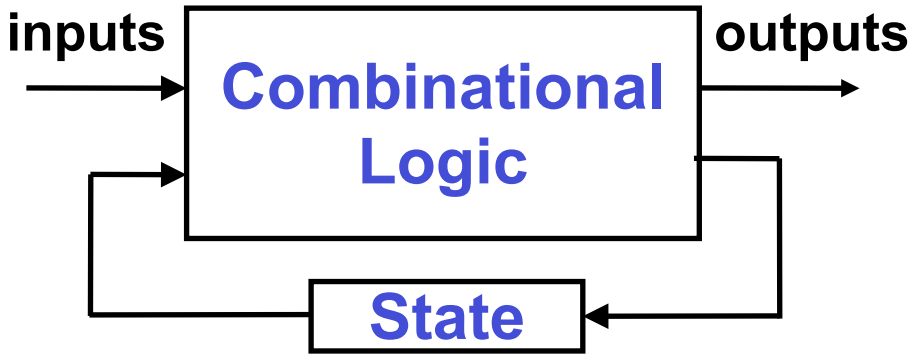
Sequential Logic:
Clocks
Latches
Flip-Flops
Counters

Cornell University

Combinational vs. Sequential Circuits

Combinational Circuit

Sequential Circuit



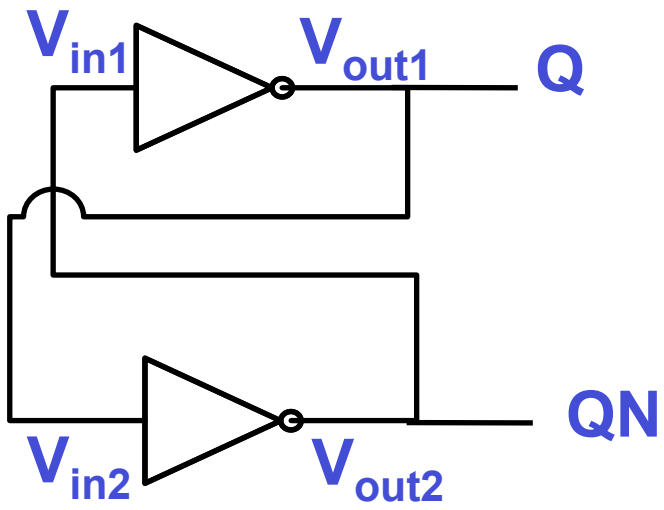
- **Combinational**
  - Output depends only on current inputs
- **Sequential**
  - Output depends on current inputs plus past history
  - Includes memory

Sequential Circuits

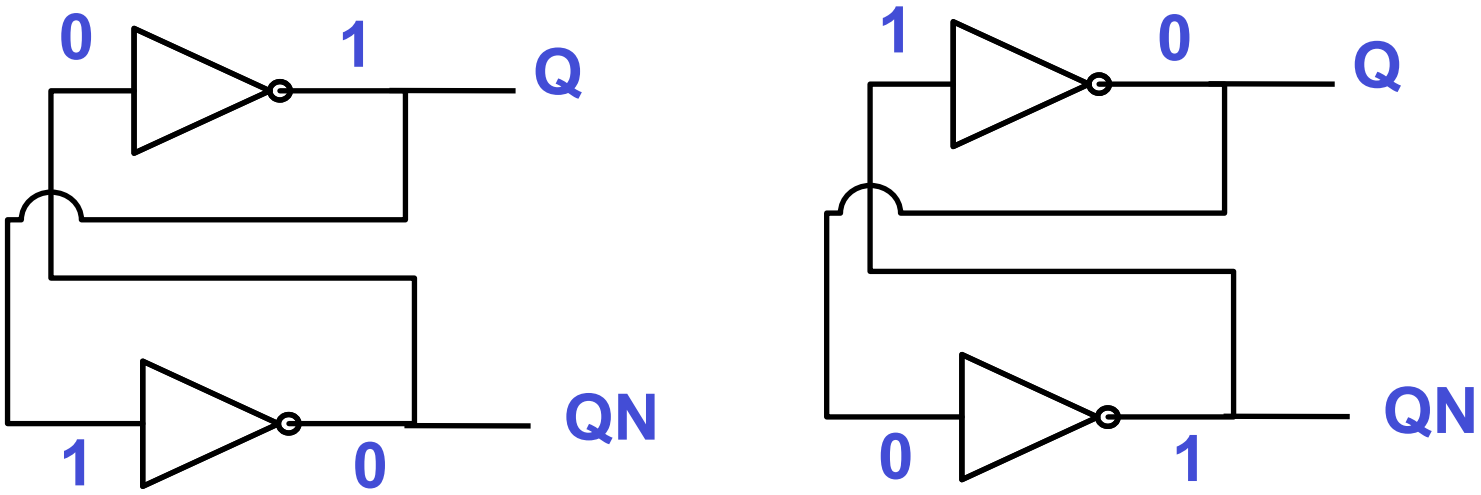
- Outputs depend on inputs and state variables
- The state variables embody the past
- Storage elements hold the state variables
- A clock periodically advances the circuit

Bistable Element

- Basic storage element
- Inverters with outputs connected to inputs

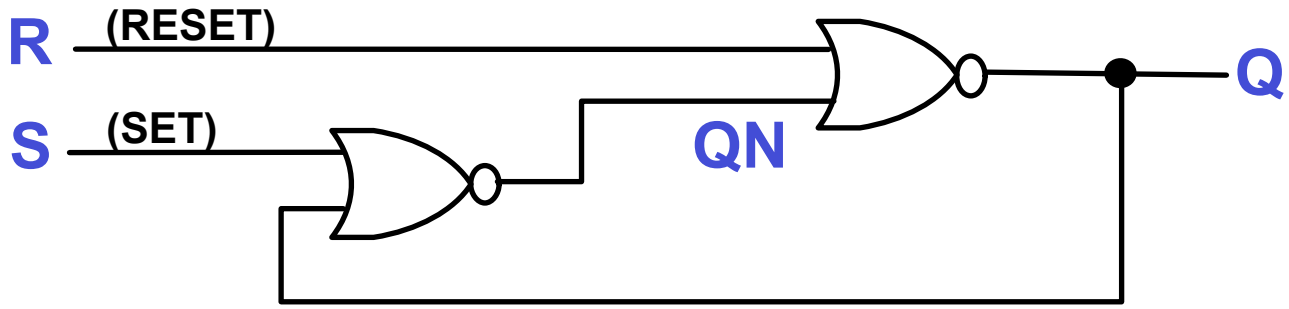


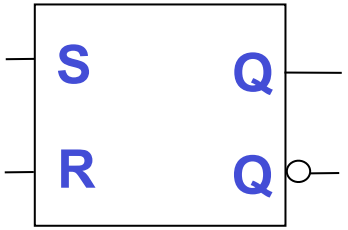
Two Stable States



Need a way to change the output

S-R Latch (Set-Reset Latch)





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

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

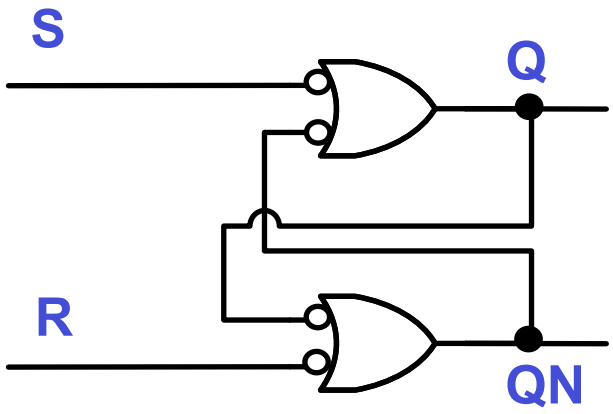
$$Q_{next} = (R + QN)'$$
$$= (R + (S + Q)')'$$
$$= R' \cdot (S + Q)$$
$$= R' \cdot S + R' \cdot Q$$

S R	R'•S R'•Q	Q_{next}
0 0	0 Q	Q
0 1	0 0	0
1 0	1 Q	1
1 1	0 0	0

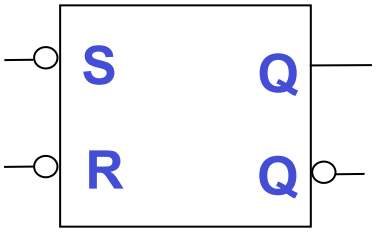
When S=0 and R=0, it holds (*latches)* its output value

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

- *S-bar-R-bar* latch
  - Built from NAND gates
  - Inputs are active low rather than active high
  - When both inputs asserted, Q = 1 rather than 0



S	R	Q	QN
0	0	1	1
0	1	1	0
1	0	0	1
1	1	Q	QN

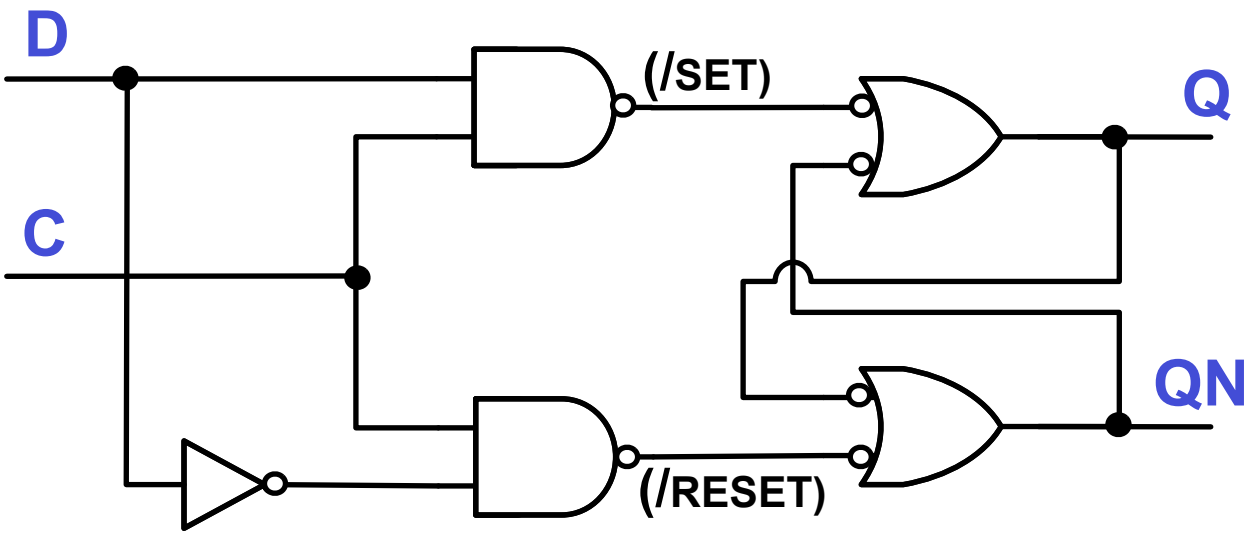


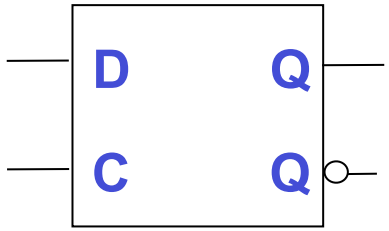
D Latch

- S-R latch *sets* or *resets* the output
- D latch *captures* input data
- Operates in two modes
  - *Open* or *transparent*: input flows through to output
  - *Closed* or *opaque*: output does not change

D Latch

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



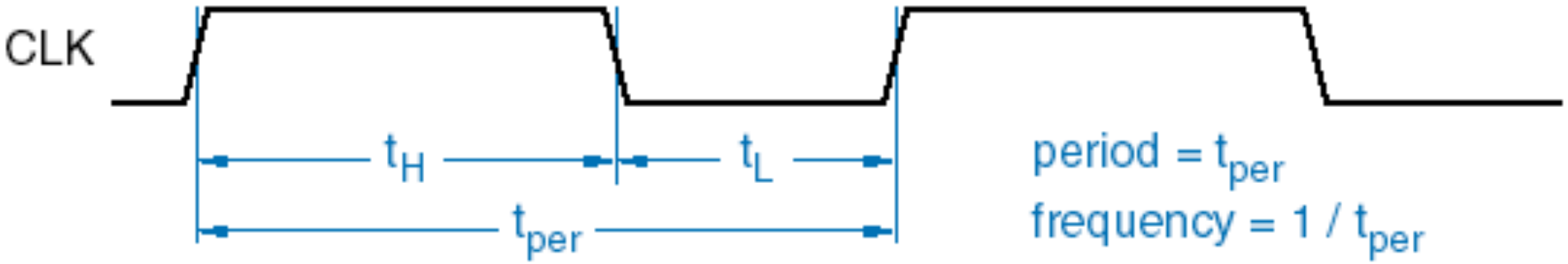


C	D	Q	QN
1	0	0	1
1	1	1	0
0	X	Last Q	Last QN

Clock

- An input to a sequential circuit that changes output and state values at a predetermined rate
- ***Clock Period***: Time between successive transitions in the same direction (L→H or H→L)
  - 1ms, 2ns, 250ps
- ***Clock Frequency***: 1/period
  - 1kHz, 500MHz, 4GHz
- ***Triggering edge***: Transition of the clock (H→L or L→H) that captures input data
- ***Clock tick***: Occurrence of a triggering edge

Clock Waveform

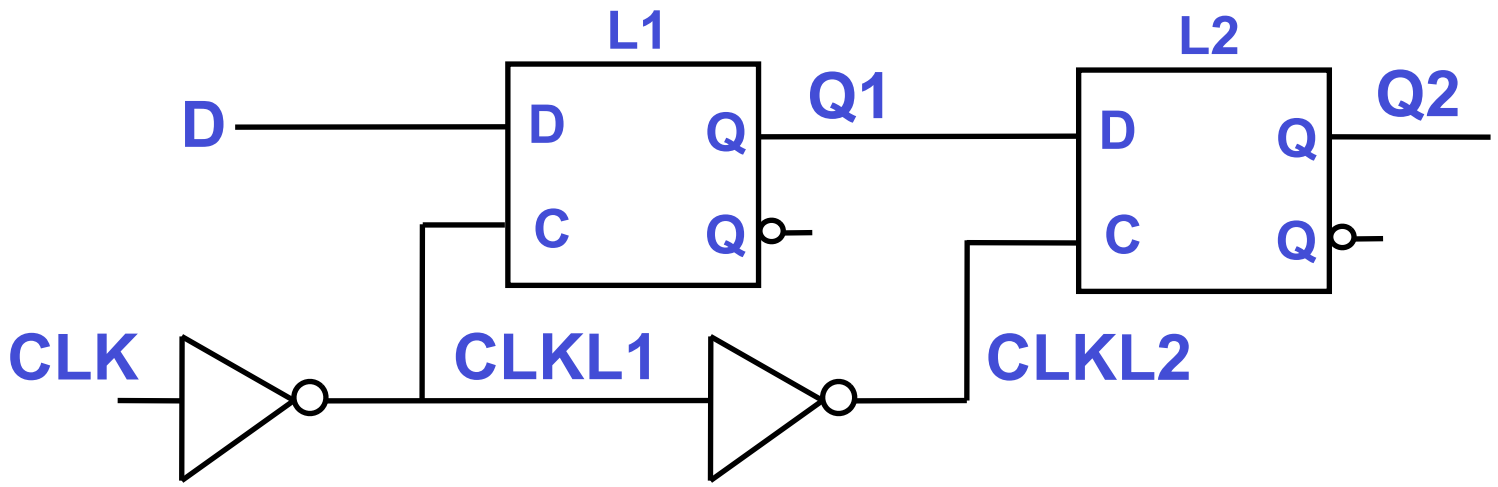


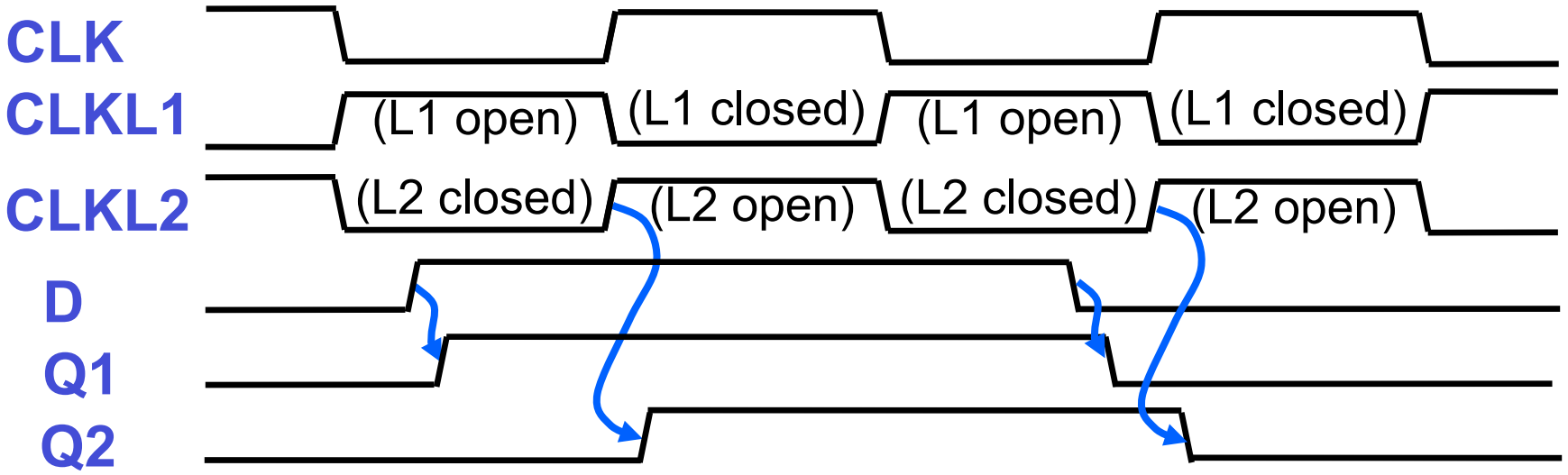
Flip-Flop

- **Samples input on *triggering edge* of clock**
  - **Rising edge → *positive edge-triggered flip-flop***
  - **Falling edge → *negative edge-triggered flip-flop***

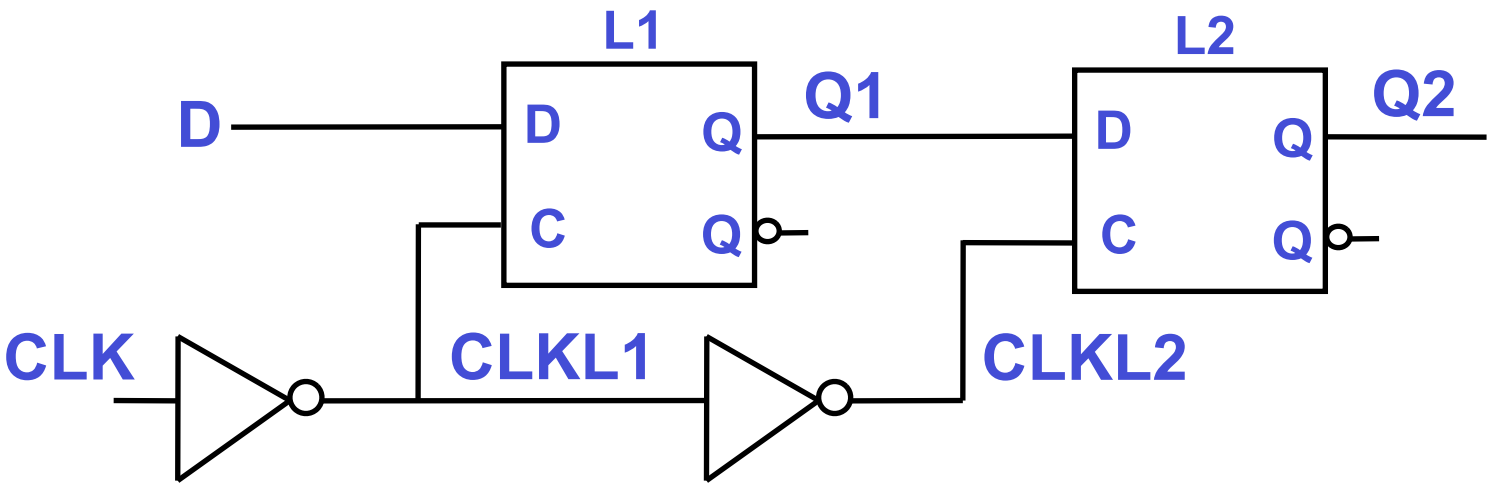
- **D flip-flop: Two D latches back-to-back**

D Flip-Flop

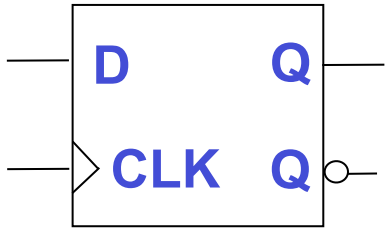

L1
L2
D
Q1
Q2
D
Q
C
Q
D
Q
C
Q
CLK
CLKL1
CLKL2


CLK
CLKL1
CLKL2
D
Q1
Q2
(L1 open)
(L1 closed)
(L1 open)
(L1 closed)
(L2 closed)
(L2 open)
(L2 closed)
(L2 open)

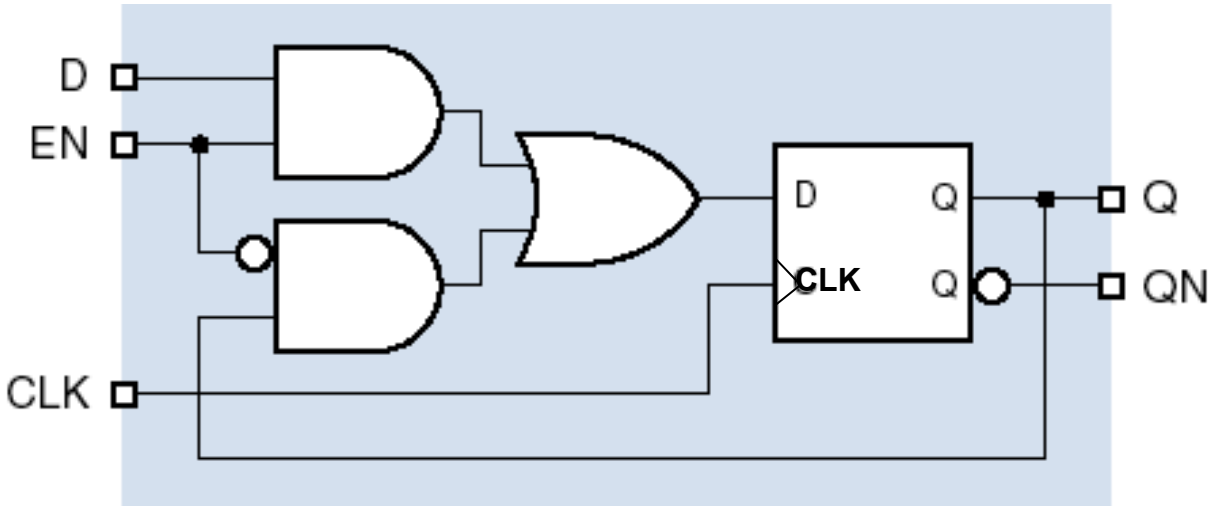
D Flip-Flop

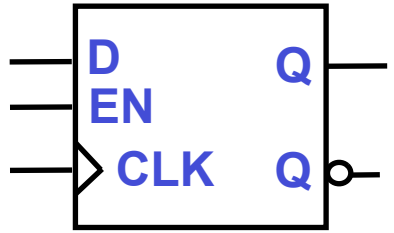


D	CLK	Q	QN
0	↑	0	1
1	↑	1	0
X	0	Last Q	Last QN
X	1	Last Q	Last QN



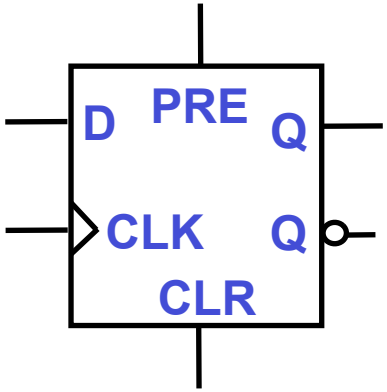
D Flip-Flop with Enable





Preset/Clear Inputs

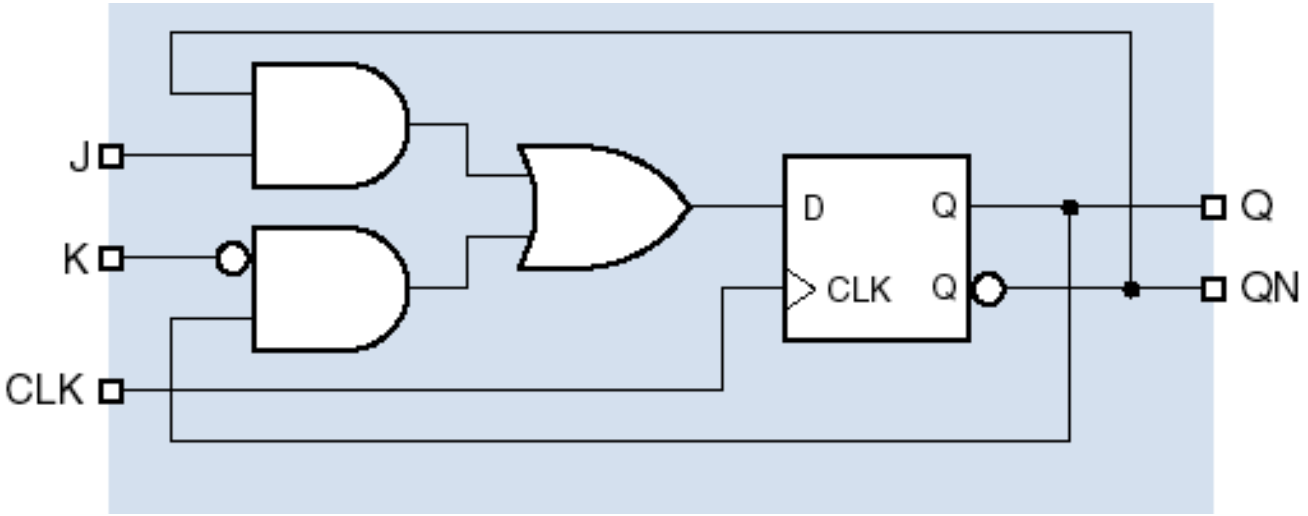
- **Preset**
  - Forces Q to 1
- **Clear**
  - Forces Q to 0

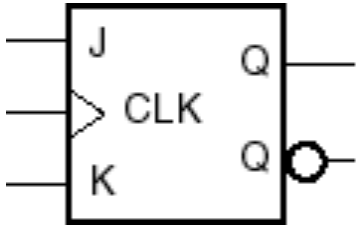


- **Asynchronous or synchronous**
  - Asynchronous: Operate independent of the clock
  - Synchronous: Only take effect on the triggering edge of the clock

J-K Flip-Flop

- A flip-flop equivalent of an S-R latch



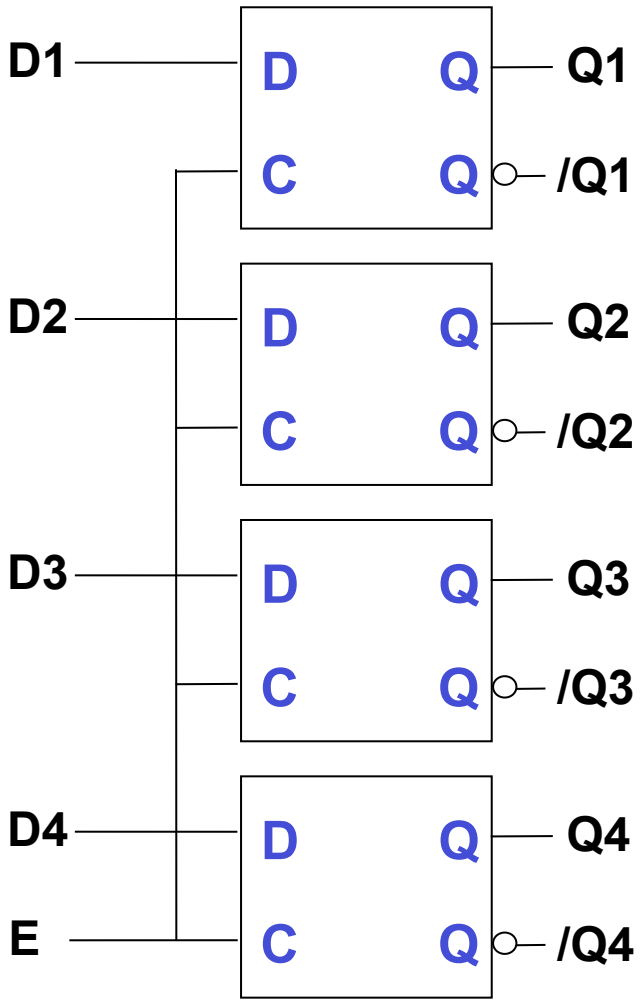


Q = J•Q' + K'•Q

J	K	CLK	Q	QN
x	x	0	last Q	last QN
x	x	1	last Q	last QN
0	0	↓	last Q	last QN
0	1	↓	0	1
1	0	↓	1	0
1	1	↓	last QN	last Q

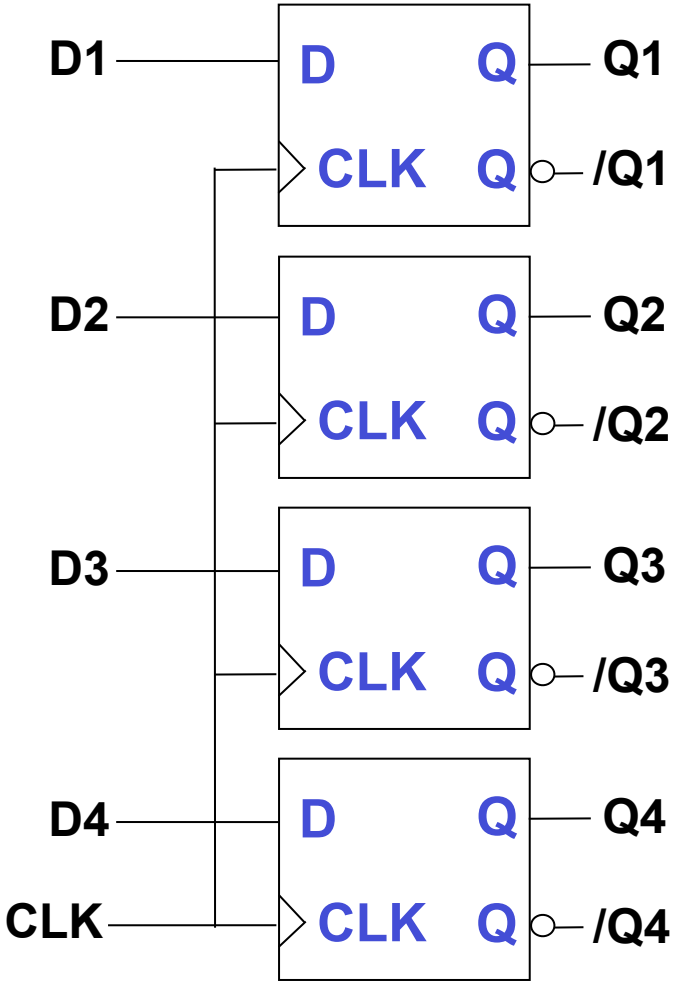
Multi-bit Latch

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



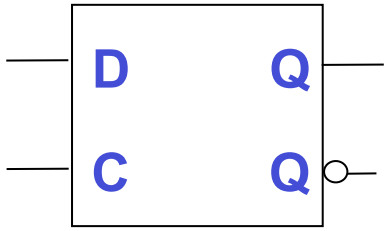
Register

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



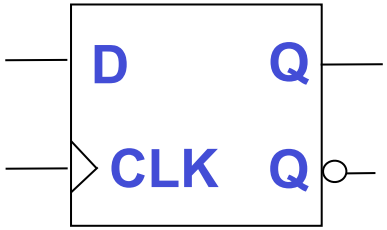
D Latch

- Output follows input when enable asserted *(open)*
- Input captured when enable deasserted *(closed)*
- Used for building memories



D Flip-Flop

- Output changes only on triggering clock edge
- Used for sequential logic design



Binary Counter

- Counts in binary in a particular sequence
- Advances at every tick of the clock
- Many types

Up	Down	Free Running	Divide-by-n	n-to-m
0 0 0	1 1 1	0 0 0	0 0 0	n
0 0 1	1 1 0	0 0 1	0 0 1	n+1
0 1 0	1 0 1	0 1 0	0 1 0	n+2
0 1 1	1 0 0	0 1 1	0 1 1	.
1 0 0	0 1 1	1 0 0	1 0 0	.
1 0 1	0 1 0	1 0 1	.	.
.	.	1 1 0	.	m-1
.	.	1 1 1	n-1	m
.	.	0 0 0	0 0 0	n
		.	0 0 1	n+1
		.		.
		.		.

Up Counter Sequence

0 0 0
0 0 1
0 1 0
0 1 1
1 0 0
1 0 1
1 1 0
1 1 1
0 0 0
0 0 1

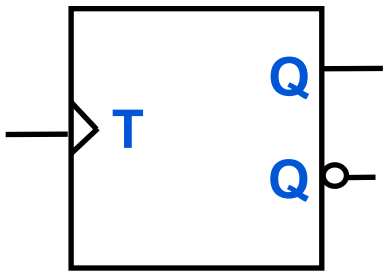
Toggles every clock tick that two right bits = 11

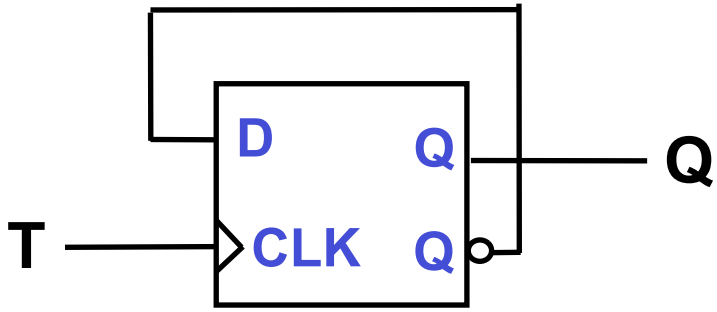
Toggles every clock tick

Toggles every clock tick that right bit = 1

T (*Toggle*) Flip-Flop

- **Output toggles each clock tick**

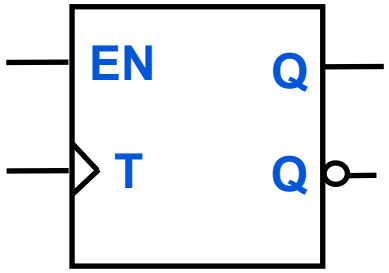




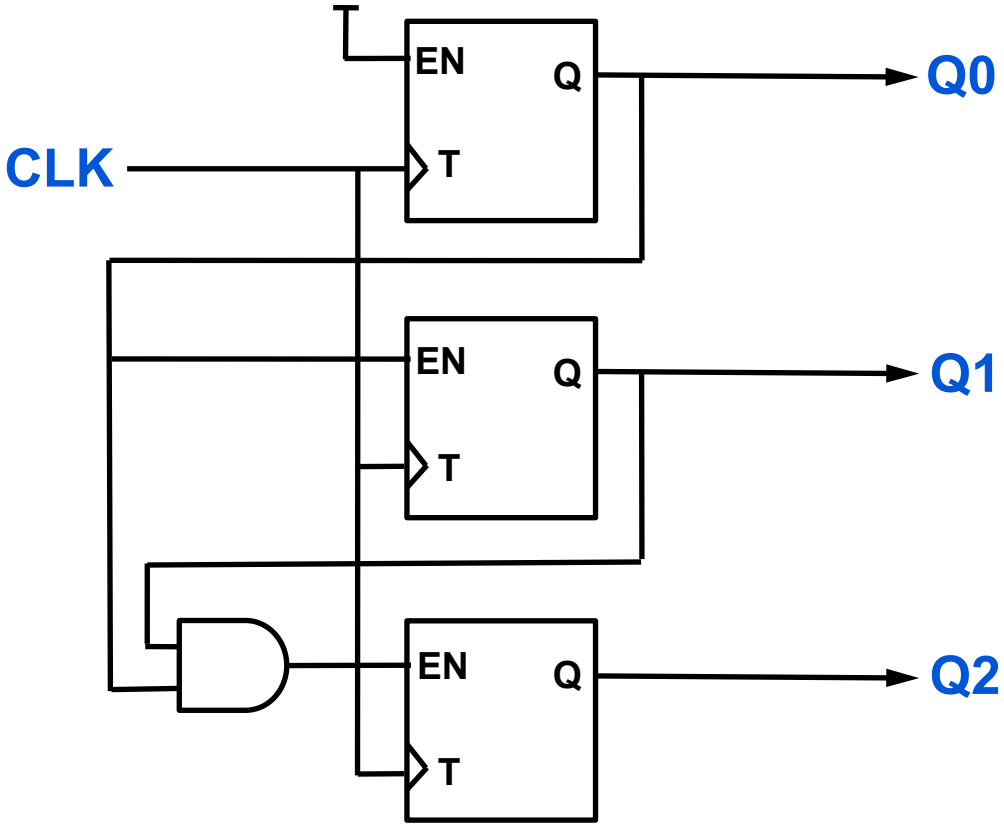
Q: 0, 1, 0, 1, 0, 1, 0, ...

T Flip-Flop with Enable

- Output toggles only if enabled (EN = 1)
- Output does not change if EN = 0

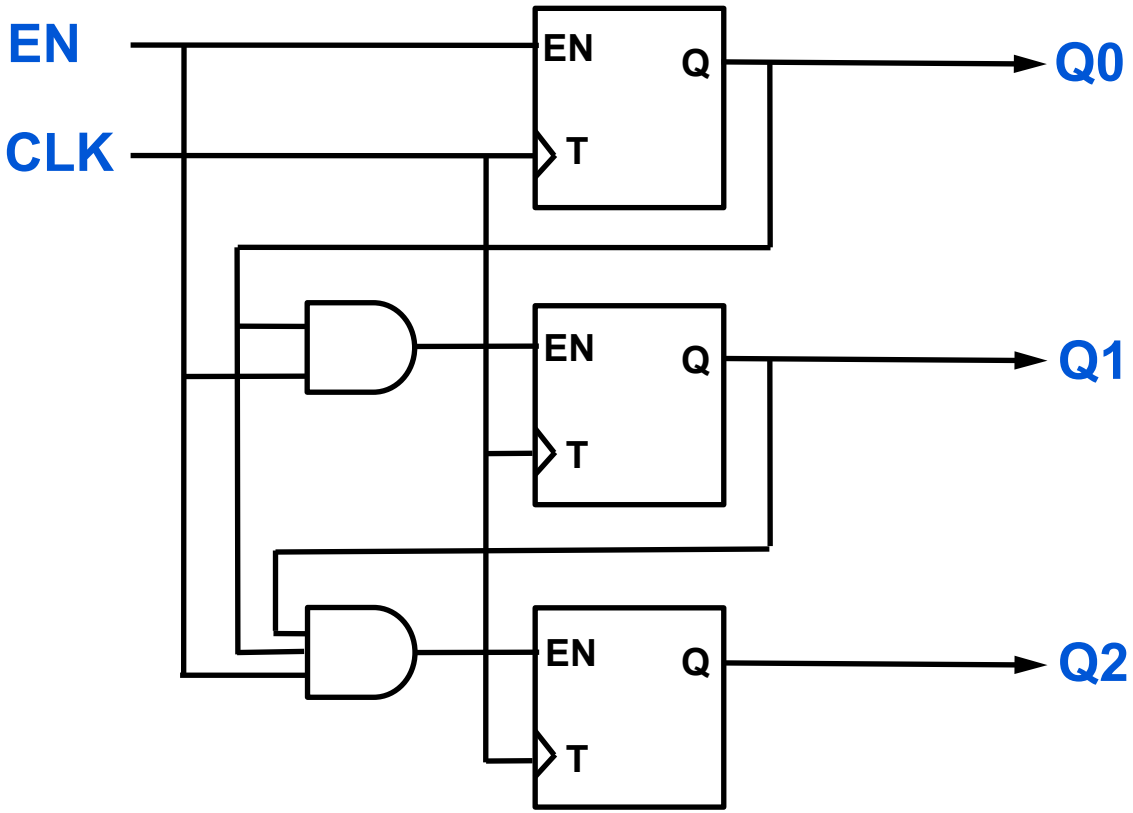


Free Running Binary Up Counter



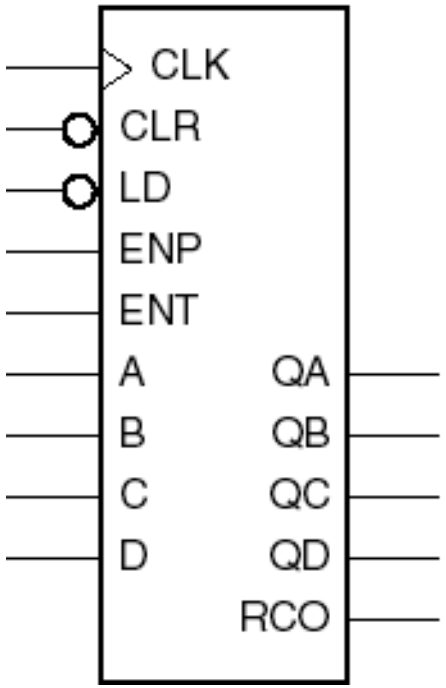
Q_2	Q_1	Q_0
0	0	0
0	0	1
0	1	0
0	1	1
1	0	0
1	0	1
1	1	0
1	1	1
0	0	0
0	0	1

Binary Up Counter with *Hold*



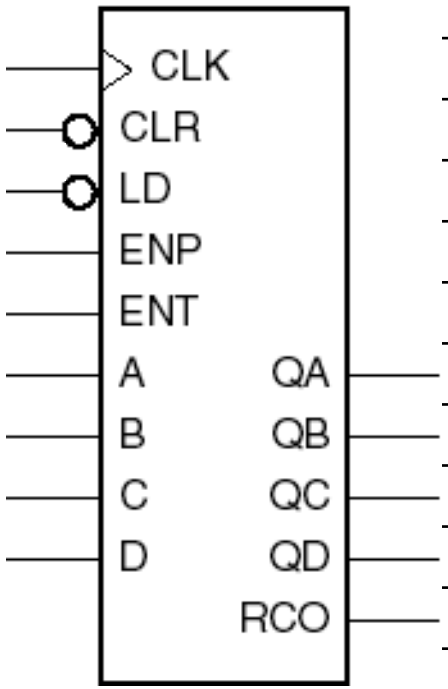
4-Bit Binary MSI Counter (’163)

- **Inputs**
  - CLK = clock input
  - CLR = synchronous clear
  - ENT AND ENP = enable counting
  - LD = load A,B,C,D (like a register)
  - A-D = load inputs



- **Outputs**
  - QA-QD = current number in sequence
  - RCO = ripple carry out
    = 1 if QA-QD = 1111, and ENT is 1

’163 Truth Table



Inputs				Current State				Next State			
/CLR	/LD	ENT	ENP	QD	QC	QB	QA	QD*	QC*	QB*	QA*
0	X	X	X	X	X	X	X	0	0	0	0
1	0	X	X	X	X	X	X	D	C	B	A
1	1	0	X	X	X	X	X	QD	QC	QB	QA
1	1	X	0	X	X	X	X	QD	QC	QB	QA
1	1	1	1	0	0	0	0	0	0	0	1
1	1	1	1	0	0	0	1	0	0	1	0
1	1	1	1	0	0	1	0	0	0	1	1
1	1	1	1	0	0	1	1	0	1	0	0
1	1	1	1	0	1	0	0	0	1	0	1
1	1	1	1	0	1	0	1	0	1	1	0
1	1	1	1	0	1	1	0	0	1	1	1
1	1	1	1	0	1	1	1	1	0	0	0
1	1	1	1	1	0	0	0	1	0	0	1
1	1	1	1	1	0	0	1	1	0	1	0
1	1	1	1	1	0	1	0	1	0	1	1
1	1	1	1	1	0	1	1	1	1	0	0
1	1	1	1	1	1	0	0	1	1	0	1
1	1	1	1	1	1	0	1	1	1	1	0
1	1	1	1	1	1	1	0	1	1	1	1
1	1	1	1	1	1	1	1	0	0	0	0

Before Next Class

- H&H 4.1-4.5 (skip VHDL parts), 5.4

Next Time

More Counters
Shift Registers
Verilog