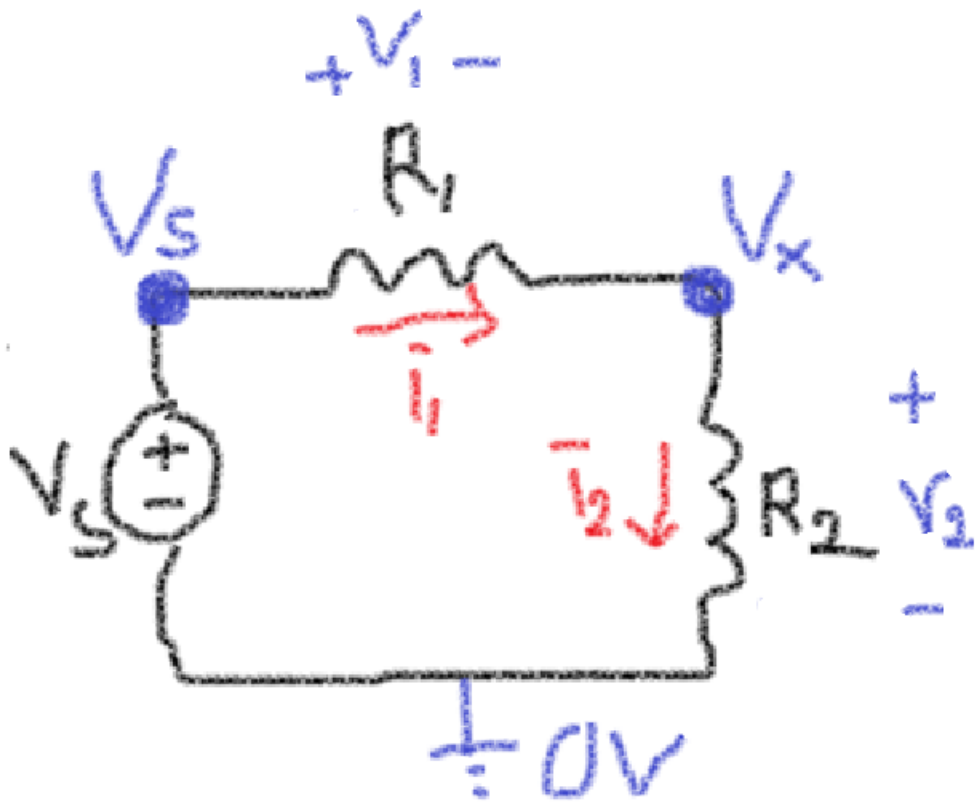
CET 141: REVIEW

Dr. Noori KIM

What we did so far…

Day 1-2: Circuit Analysis Techniques

- Method 1: Basic KVL, KCL method of Circuit analysis
- Method 2: Element combination rules
- Method 3: Node analysis
- Method 4: Superposition
- Method 5: The Thévenin Method
- Method 6: The Norton Method

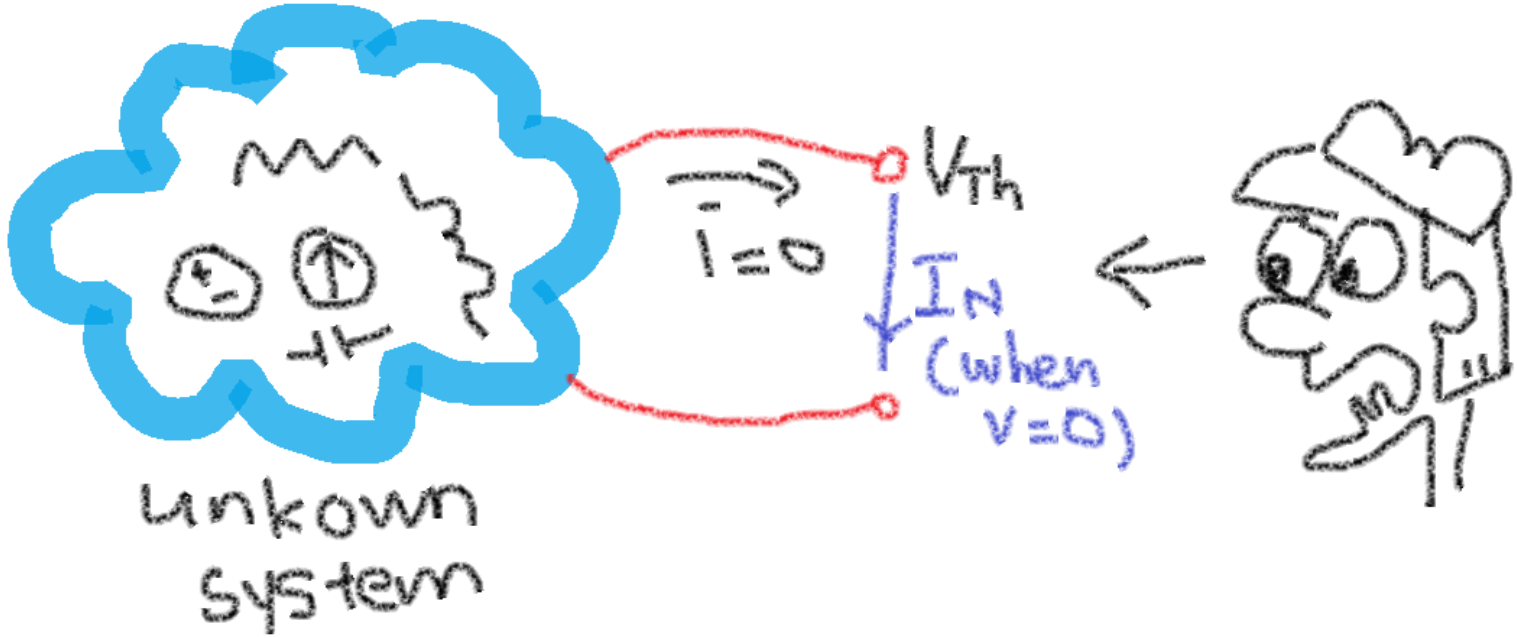


- Ohm's law
- KCL/KVL
- Nodal analysis

- Ohm's law:
  V1=R1*i1, V2=R2*i2
- KVL: Vs=V1+V2
- KCL: i1=i2 (=i)
- Nodal analysis:
  (Vs-Vx)/R1=(Vx-0)/R2

Note that v1=Vs-Vx
i=(R1+R2)/Vs=i1=V1/R1=i2=V2/R2

- Thevenin/Norton equivalents ⇨ Source transformation



R_{th}

- Only independent sources: V short, I open
- With dependent sources: take a ratio (V_{th}/I_N)

VIF

$$V = IR$$

KVL: Gain=Drop

KCL: In=Out

C C G G A A G F F E E D D C

- Open Circuit I is zero, R is infty, as like me.

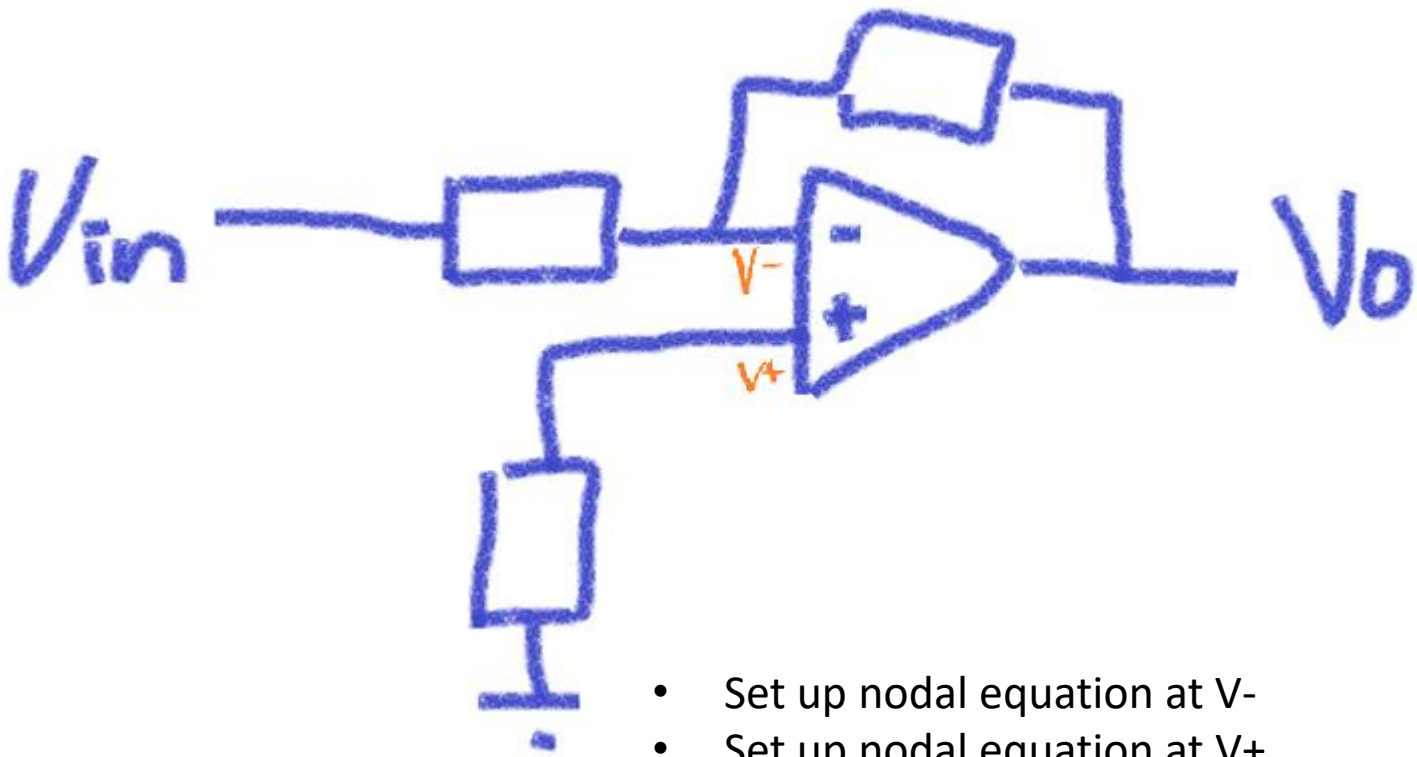
G G F F E E D G G F F E E D

- Short Circuit V is zero, R is zero, like a wire

C C G G A A G F F E E D D C

- Open circuit I is zero, Short Circuit V is zero

Day 3-4: Adding op-amps on our circuits & Quiz1



- Set up nodal equation at V-
- Set up nodal equation at V+
- Set V-=V+

Day 5: L and C components

VIF

$$i_c(t) = C\frac{d}{dt}v_c(t), \text{ then } v_c(t)?$$

$$v_L(t) = L\frac{d}{dt}i_L(t), \text{ then } i_L(t)?$$

Day 6: RC/RL circuits

- We add C and L into our circuits one thing at a time
  - Solve for Liner 1st order differential equations
    - $X = v(t) + Y\frac{d}{dt}v(t)$, solve for v(t)
  - Still DC sources
- Formal way: Complete sol= Homogeneous sol+ particular sol
- Quick way: $K1+K2e^{-t/\tau}$

VIF

$$v_c(t) = K_1 + K_2 e^{-\frac{t}{\tau}}$$

$$i_L(t) = K_1 + K_2 e^{-\frac{t}{\tau}}$$

Where $\tau = \mathbf{R_{Th}}C$ or $L/\mathbf{R_{Th}}$

For RC circuits, get V_c first
For RL circuits, get i_L first

Day 7-8: Frequency domain circuit analysis, Phasor &&Quiz2

- AC inputs to LCR components
- Observe voltage and current waves' changing
  - Only phase shifts happen in L and C cases
  - Phasor analysis is useful as frequency is not changed
- Come back and forth between F and T domains

VIF

Euler's formula: $\boldsymbol{e^{jx}} = \cos \boldsymbol{x} + \mathrm{j}\sin \boldsymbol{x}$

$$\sin(\omega t) = \cos(\omega t - 90^o)$$

- The concept of impedance for R, L, C in frequency domain: **Z=R+iX (Ω)**
  - R: resistance (Real part), X: reactance (Imag part)

VIF

$$Z_L = j\omega L$$
$$Z_C = 1/(j\omega C)$$
$$Z_R = R$$

- Same circuit analysis technique applies in the frequency domain
  - KCL, KVL, nodal/mesh analysis

More interesting things: Matrix circuit analysis

- Strengthen and solidify your circuit analysis skills
- ABCD matrix and Z matrix

Computation of ABCD parameters

1. Take each component
2. Calculate Z (impedance) of the component
   - R, jωL, 1/(jωC)
3. A single matrix for a series component(s)
   - $\begin{bmatrix} A & B \\ C & D \end{bmatrix} = \begin{bmatrix} 1 & Z \\ 0 & 1 \end{bmatrix}$
4. A single matrix for a shunt components(s)
   - $\begin{bmatrix} A & B \\ C & D \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ \frac{1}{Z} & 1 \end{bmatrix}$
5. Cascade all components' matrix

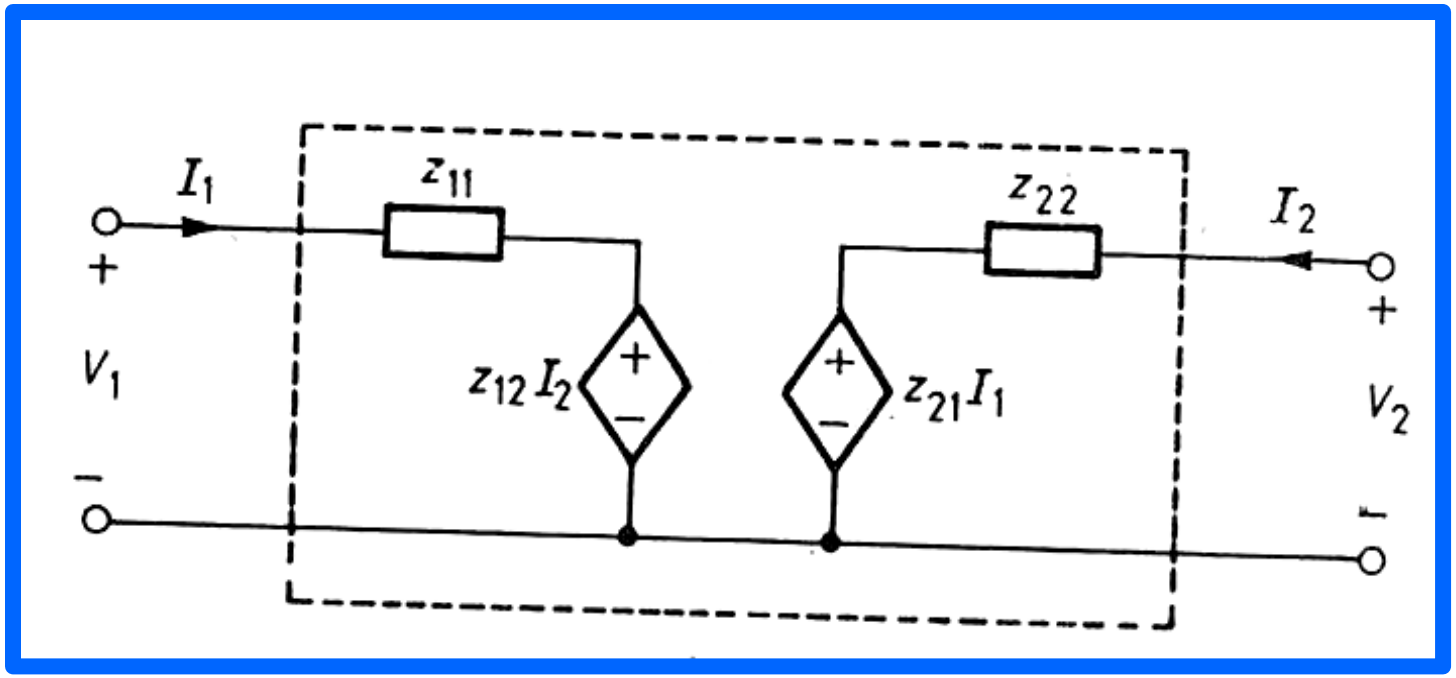
$$\begin{bmatrix} z_{11} & z_{12} \\ z_{21} & z_{22} \end{bmatrix} \leftarrow \begin{bmatrix} \frac{A}{C} & \frac{\Delta_T}{C} \\ \frac{1}{C} & \frac{D}{C} \end{bmatrix}$$

$$\uparrow$$

$$\begin{bmatrix} A & B \\ C & D \end{bmatrix}$$

Equivalent Circuit Using *z*-parameters

It is just a picturization for defining equations.



$$z_{11} = \frac{V_1}{I_1}\Bigg|_{I_2=0}$$

$$z_{12} = \frac{V_1}{I_2}\Bigg|_{I_1=0}$$

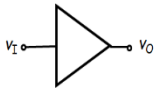
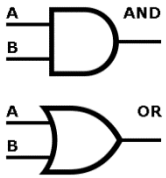
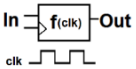
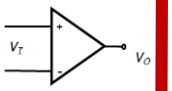
$$z_{21} = \frac{V_2}{I_1}\Bigg|_{I_2=0}$$

$$z_{22} = \frac{V_2}{I_2}\Bigg|_{I_1=0}$$

$$V_1 = z_{11} I_1 + z_{12} I_2$$

$$V_2 = z_{21} I_1 + z_{22} I_2$$

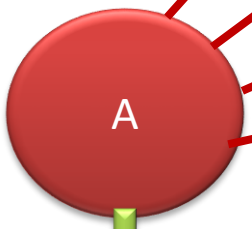
ECE Layers by Abstraction

Natures	Physics laws	Lumped circuit Abst.	Amplifier		Logic gates	Combinational logic	Clocked Digital Abst.	Instruction set Architecture (ISA)	Computer language	**SW Abst. starts**
Measuring using probes V I 3 0.1 6 0.2 9 0.3	V= IR Maxwell Eqs.	 R  C  Voltage source **ECE210**	 More interesting components without considering MEs	Digital	 (elementary bldg. blocks to inverters)	Define functional blocks 		Machine language, i.e.,X86 Instruction set (to build up micro processors)	JAVA C C++	Operating systems i.e., Linux Windows
				Analog	Op-amp	Analog system components	Fun devices	☺$$		Useful to human beings
						Oscillator Filter Power supply	Toasters, Power plants	☺ $		Video games, Space ships ...

ABSTRACTION!!!! From natures, all the way to devices
Bigger things, complicate behavior inside, but simple to describe

Day 9: Digital circuits Intro

Mostly, Digital and Analog components coexist



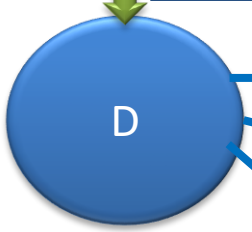
Linear elements such as C, L, R

2 terminal devices

Continuous levels

Basic building blk: Op-amps

The invention of transistors (3 terminal devices)



Non linear elements

0 or 1: Two levels

Basic building blk: Logic gates

Examples

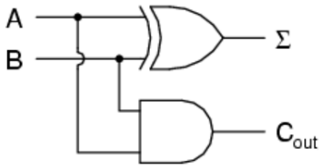
Not gates (inverters)

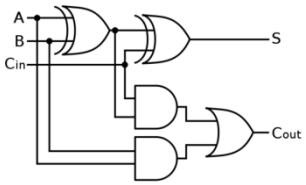
2* Not =Latch

2* Latch =F.F.

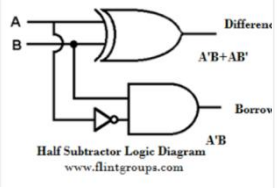
Memory & Register

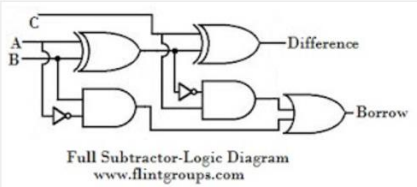
Adders



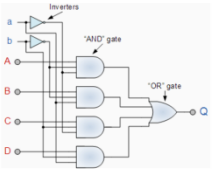


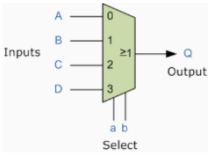
Subtractors





Multiplexers





.....

Mostly, Digital and Analog components coexist

Linear elements such as C, L, R

2 terminal devices

Continuous levels

A

Basic building blk: Op-amps

The invention of transistors (3 terminal devices)

D

Non linear elements

0 or 1: Two levels

Basic building blk: Logic gates

Clock (Memorization)

Examples

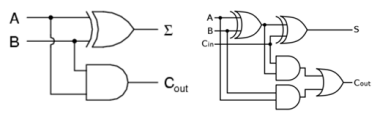
Not gates (inverters)

2* Not =Latch

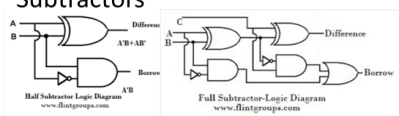
2* Latch =F.F.

Memory & Register

Adders



Subtractors



Multiplexers

.....

Sequential Logics (SL)

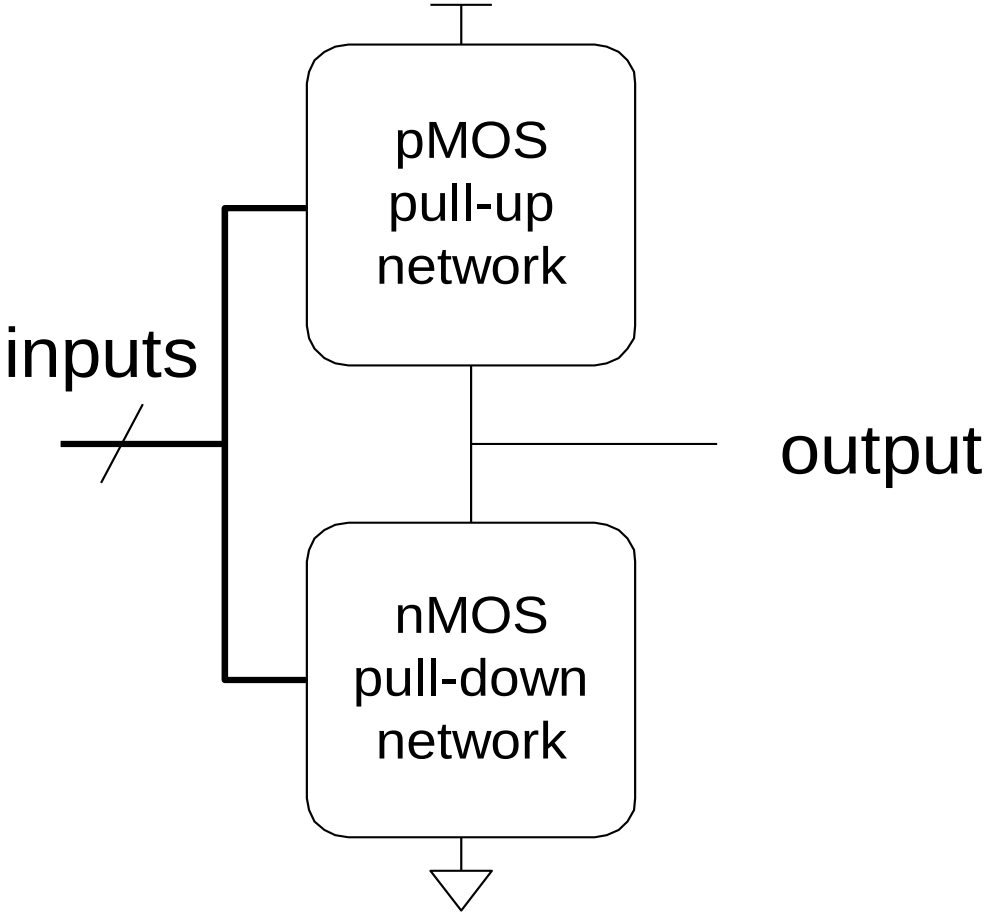
- Things are in-order
- Tools to be used to understand SL:
  - FSM (Mealy, Moore)
  - Synchronous/Asynchronous concepts
  - SL Delay analysis

Combinational Logics (CL)

- Tools to be used to understand both CL & SL:
  - Truth table, K-map, SOP, POS, &
  - Analyzing them in transistor levels (i.e., CMOS)
  - CL Delay analysis

Diode -> Transistor (BJT, CMOS) -> Logic gate (not) -> Latch -> Flip flop -> Memory and register

Keep in your mind: CMOS Gate Structure



Mapping a Standard POS Expression

$$(A+B+C)(A+B+\overline{C})(A+\overline{B}+C)(A+\overline{B}+\overline{C})(\overline{A}+\overline{B}+C)$$

AB \ C	0	1
00	0	0
01	0	0
11	0	1
10	1	1

A

$\overline{B}+C$

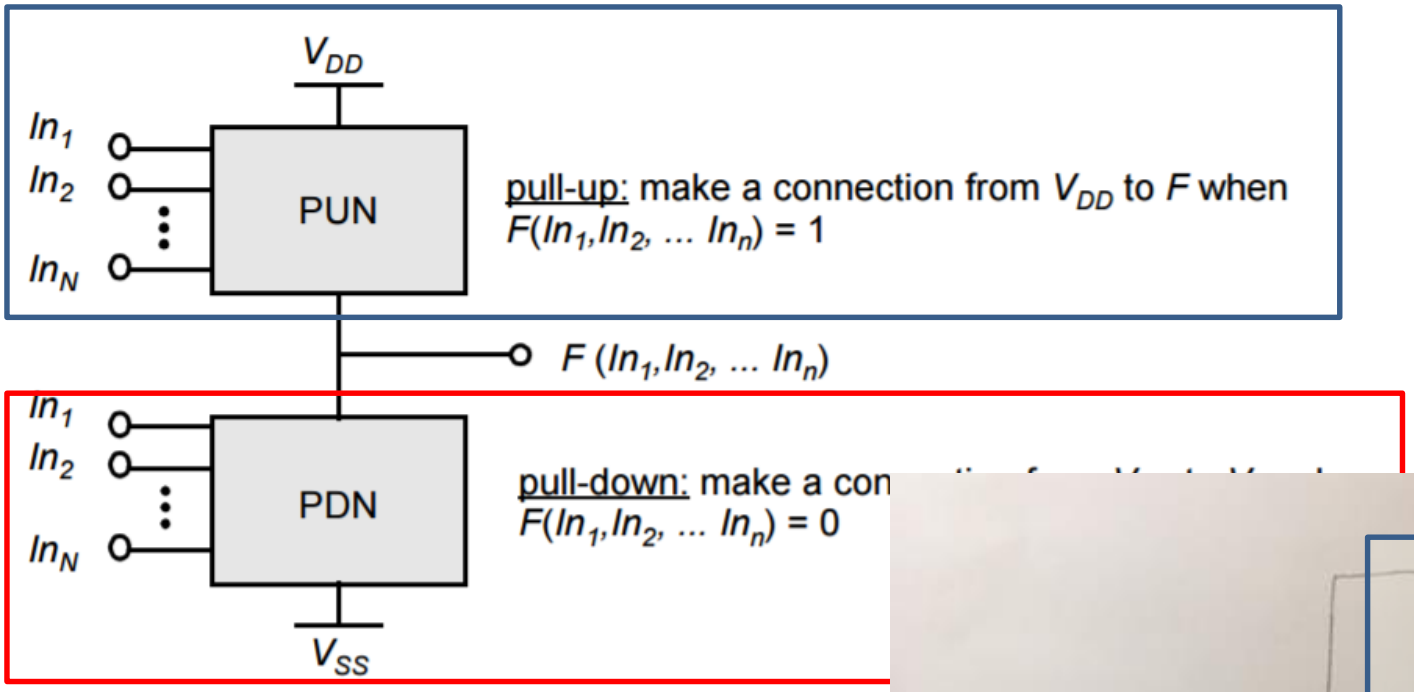
AC

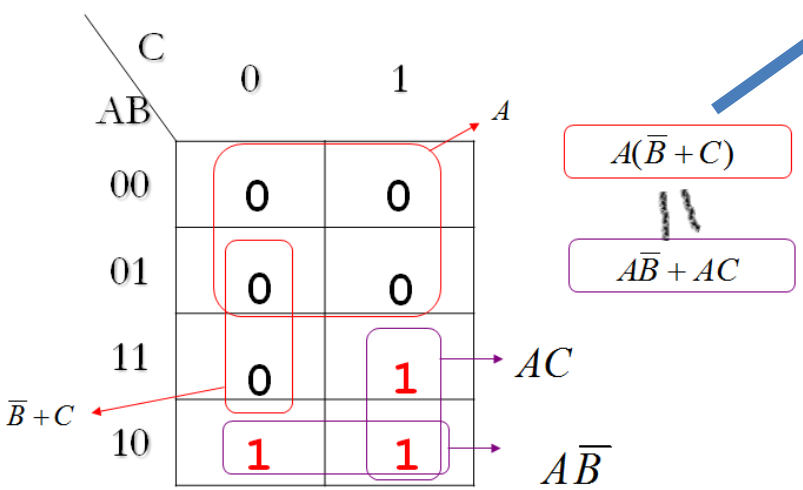
$A\overline{B}$

$A(\overline{B}+C)$

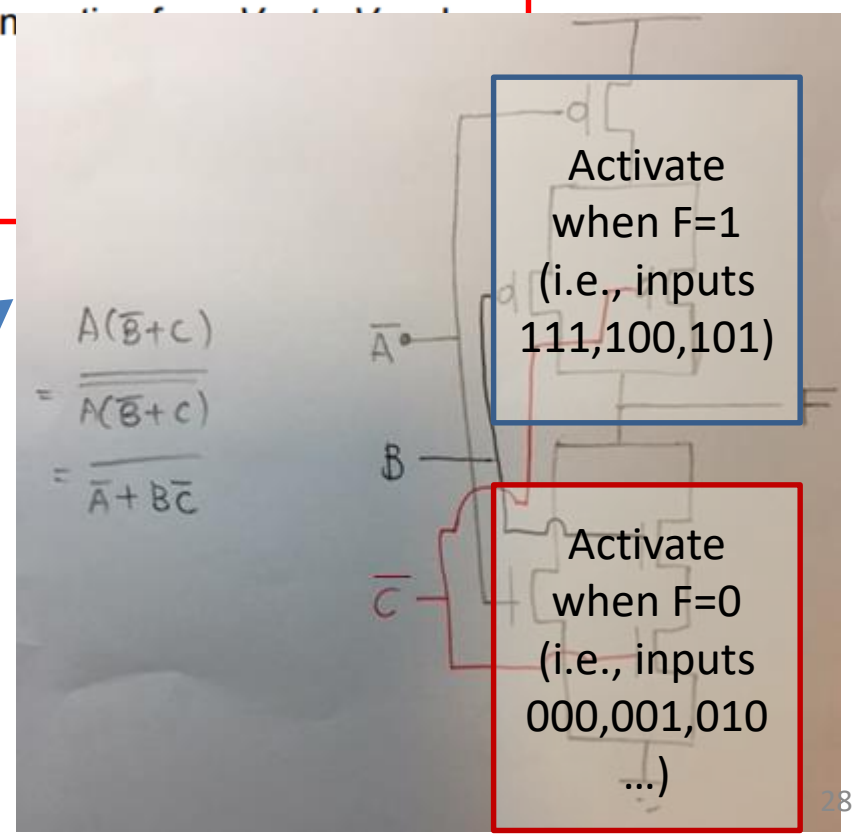
$A\overline{B}+AC$

We will see this later again, but for now.. Let's remind ourselves….





$$A(\overline{B}+C)$$
$$\parallel$$
$$A\overline{B}+AC$$



One level higher from transistors

1. Boolean algebra
2. Concept of POS/SOP (non-standard, standard)
3. K-map drawing and grouping
4. POS/SOP construction from K-map
5. POS/SOP Simplification (combinational logic functions)
6. Implement logic functions using CMOS transistors (PUN/PDN)

Implement your combinational logic functions using logic gates!!! You assume that there are lots of transistors in each logic gate unit that you use

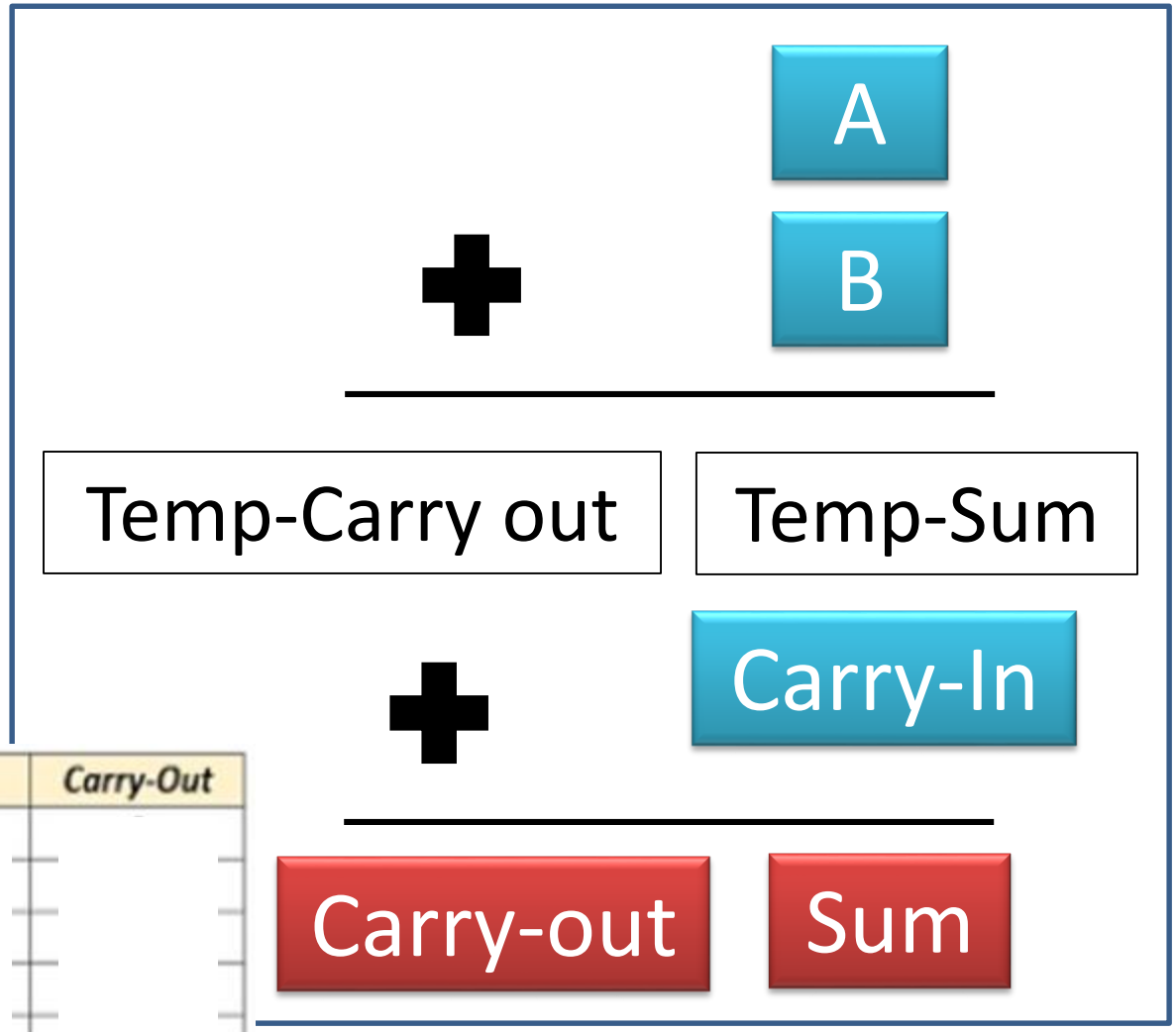
Day 10-12: Comb logic application && Quiz 3

- Lecture
  - Adder/Subtractor
  - Encoder/Decoder
  - Multiplexer/Demultiplexer
  - ALU

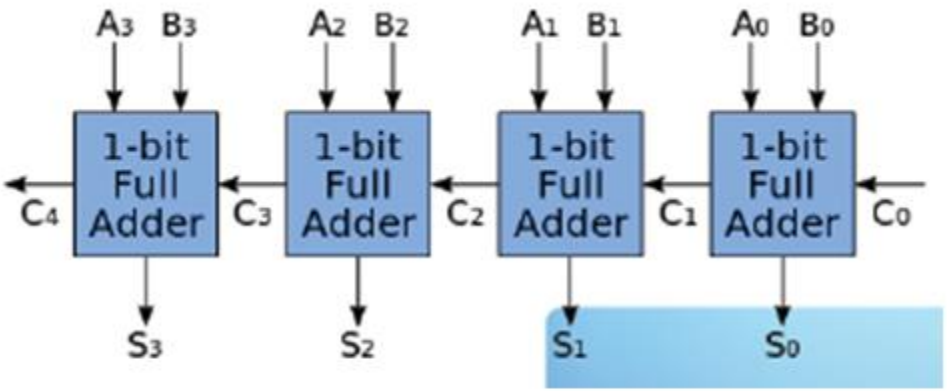
Full adder

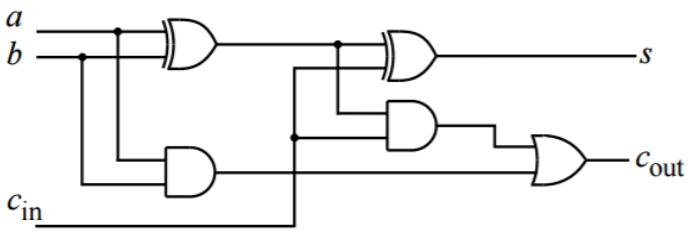
Inputs

Outputs



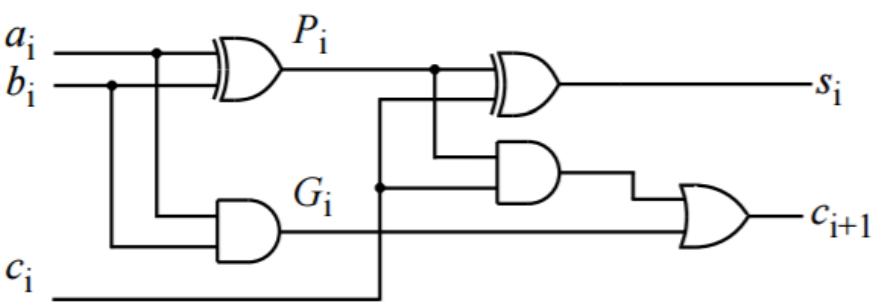
A	B	Carry-In	Sum	Carry-Out
0	0	0		
0	0	1		
0	1	0		
0	1	1		
1	0	0		
1	0	1		
1	1	0		
1	1	1		

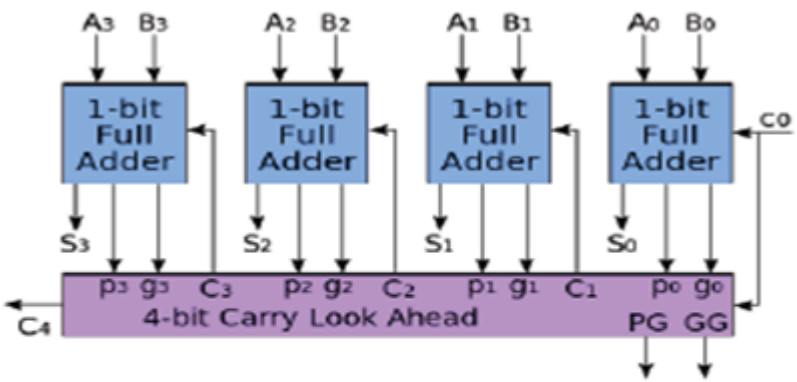




$$S_i = A_i \oplus B_i \oplus C_i$$
$$C_{i+1} = A_i B_i + C_i(A_i \oplus B_i)$$

Delay for carry: O(n)



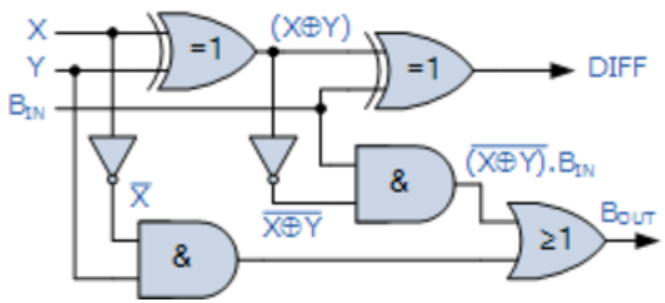


To calculate the carry signals

- No need to wait for the carry to ripple through all the previous stages to find its proper value

$$S_i = P_i \oplus C_i \qquad P_i = A_i \oplus B_i$$
$$C_{i+1} = G_i + P_i C_i \qquad G_i = A_i \bullet B_i$$

Delay for carry: O(log(n))

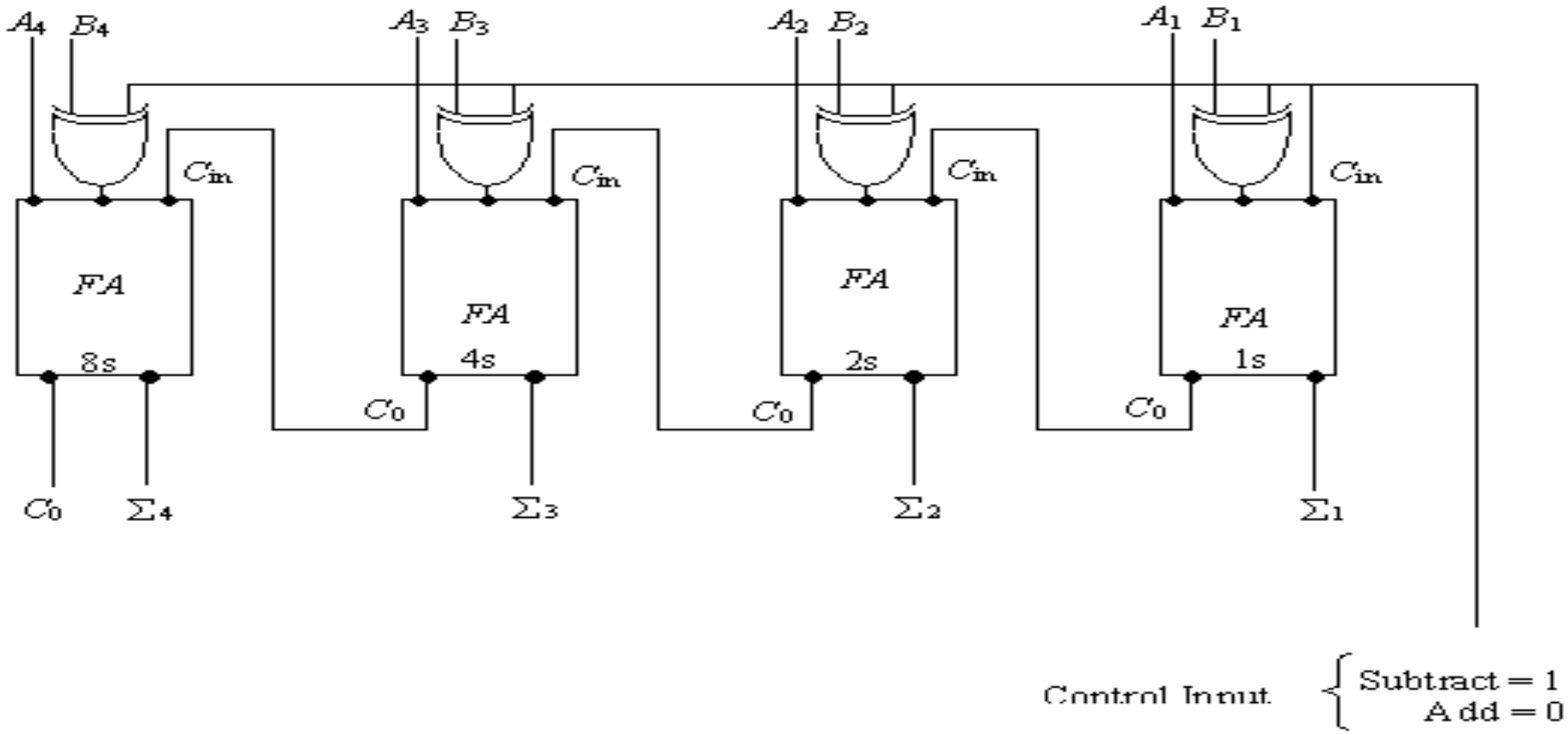
Symbol	Truth Table				
	B-in	Y	X	Diff.	B-out
	0	0	0	Try it!	
	0	0	1		
	0	1	0		
	0	1	1		
	1	0	0		
	1	0	1		
	1	1	0		
	1	1	1		

$$Di = A \oplus B \oplus Bin$$

$$B_0 = \overline{A} \cdot B + \overline{A \oplus B} \cdot B_{IN}$$

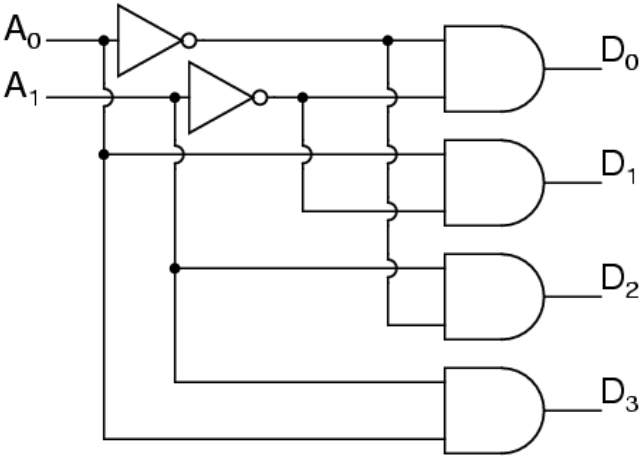
Do K-map for verification

4 bit Parallel Adder / Subtractor Circuit



The 2-to-4 line decoder

A_1	A_0	D_3	D_2	D_1	D_0
0	0				
0	1				
1	0				
1	1				



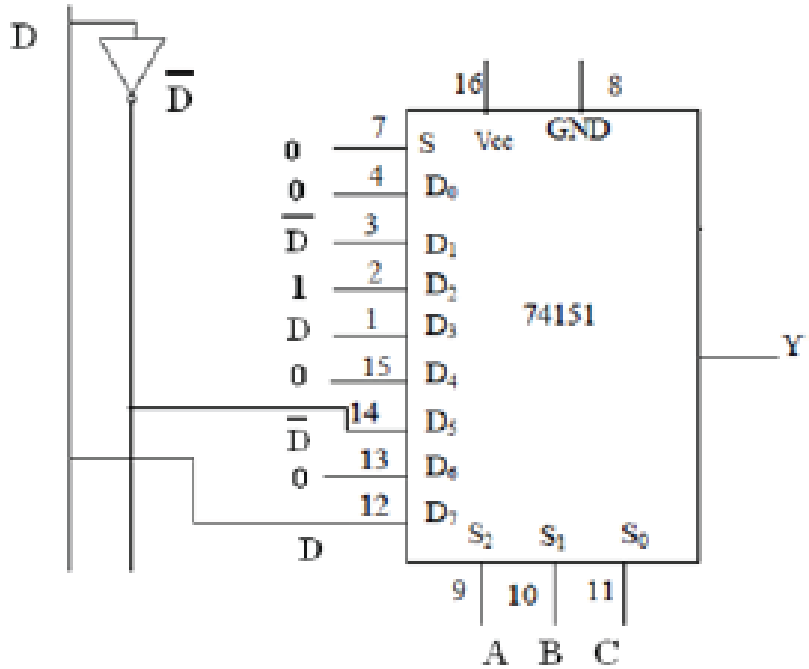
- A typical application of a line decoder circuit is to select among multiple devices.

4-to-2 **Encoder**

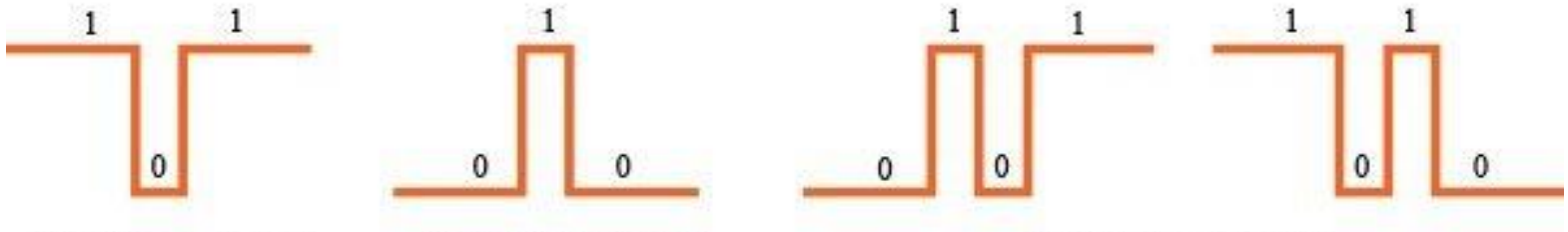
F (A, B, C, D) = ∑m (2, 4, 5, 7, 10, 14)

Design Using MSB Bit D:

	$\overline{D}$	D	
D0	0	1	0
D1	2	3	$\overline{D}$
D2	4	5	1
D3	6	7	D
D4	8	9	0
D5	10	11	$\overline{D}$
D6	12	13	0
D7	14	15	$\overline{D}$



Types of hazard

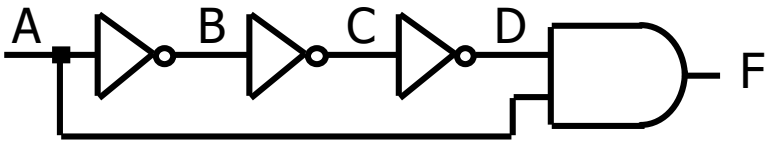


(a) Static 1- hazard (b) Static 0- hazard (c) Dynamic hazards

- Static:
  - 1 hazard: output momentarily goes to 0 when it should remain a constant value of 1
  - 0 hazard: output momentarily goes to 1 when it should remain a constant value of 0
- Dynamic
  - an output may change three or more times

Timing diagrams

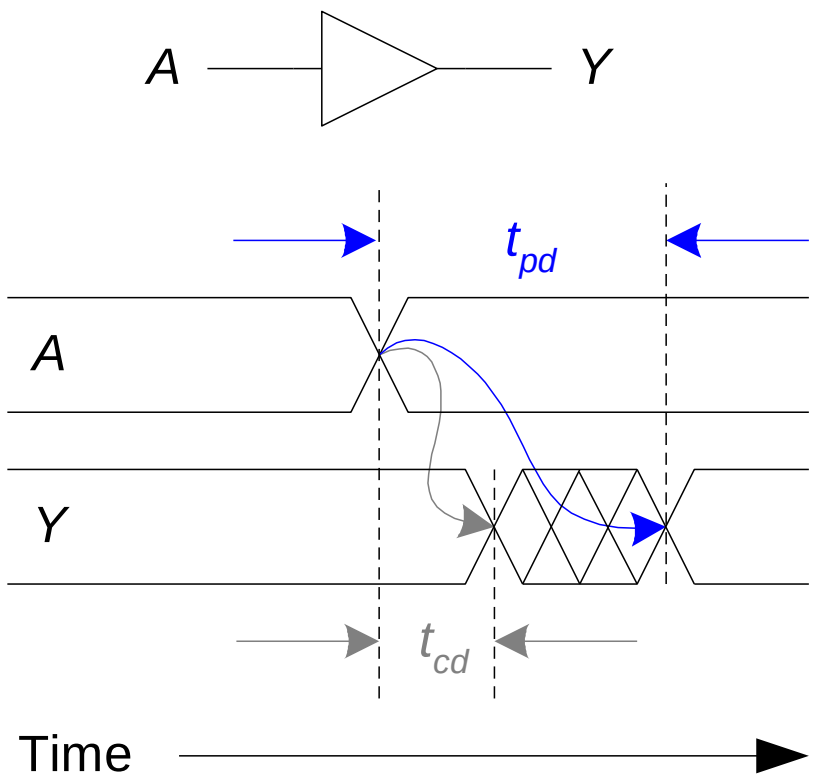
- The idea starts from: **<u>Real gates</u>** have **<u>real delays</u>**
- Example: A’’’ • A = 0?



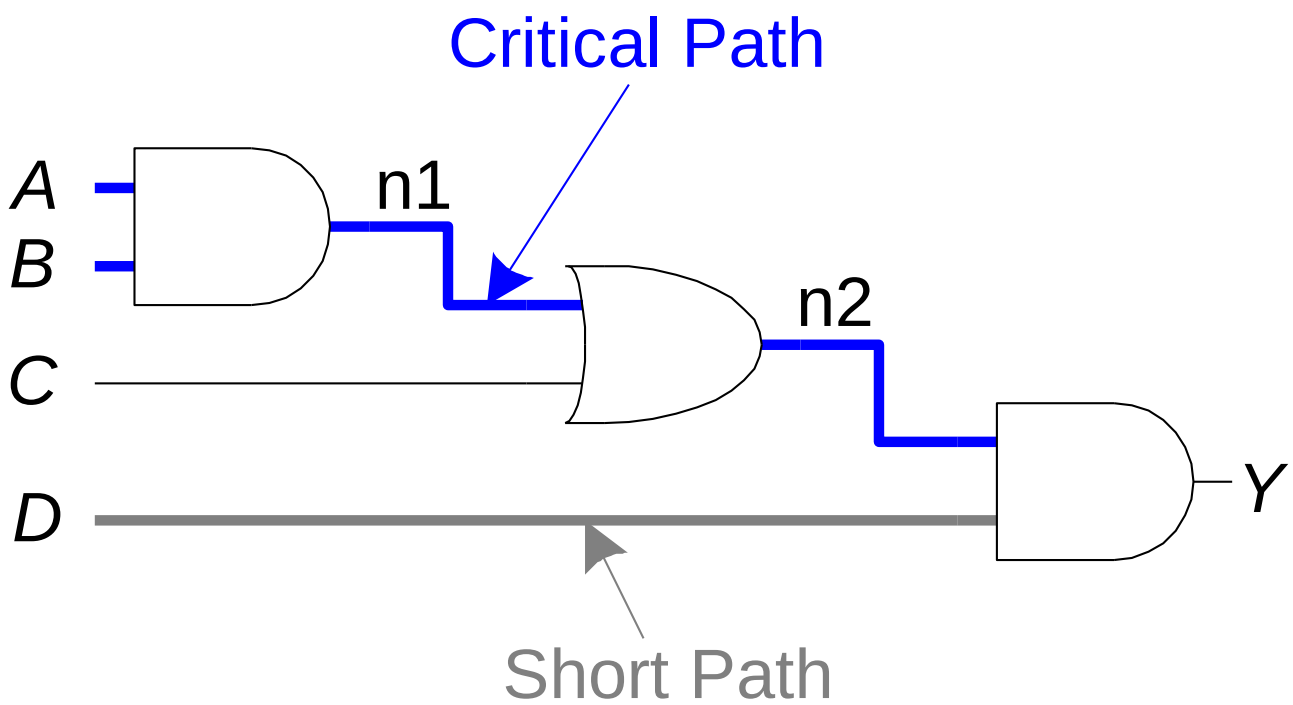
- Delays cause Glitch (F supposes to be 0 but momentarily it stays as 1: glitches): **“Static 0 hazard”**

Propagation & Contamination Delay

- **Propagation delay:** t_{pd} = max delay from input to output
- **Contamination delay:** t_{cd} = min delay from input to output

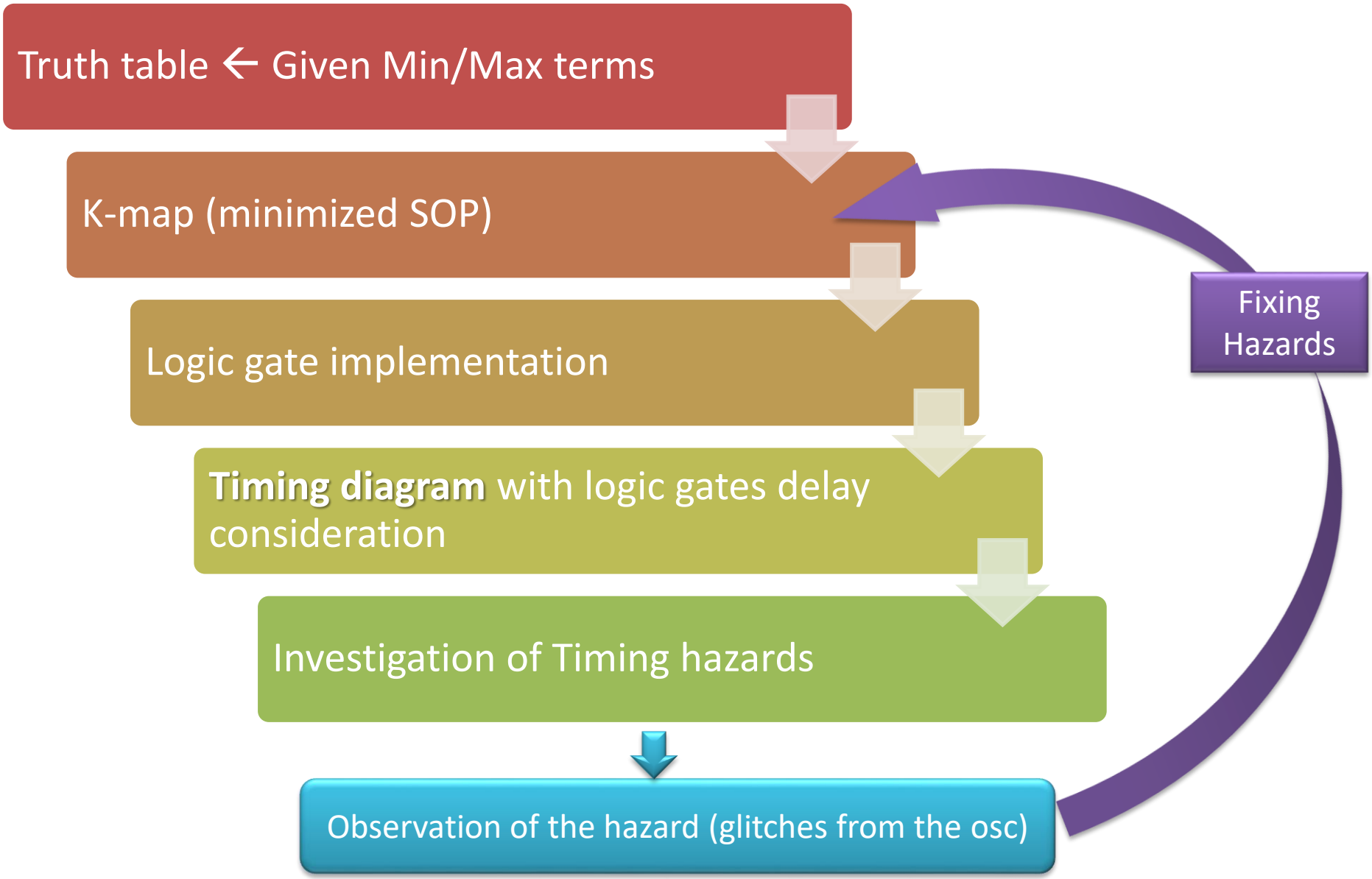


Critical (Long) & Short Paths

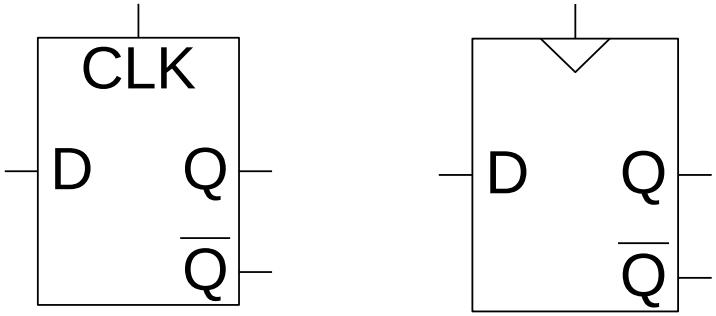


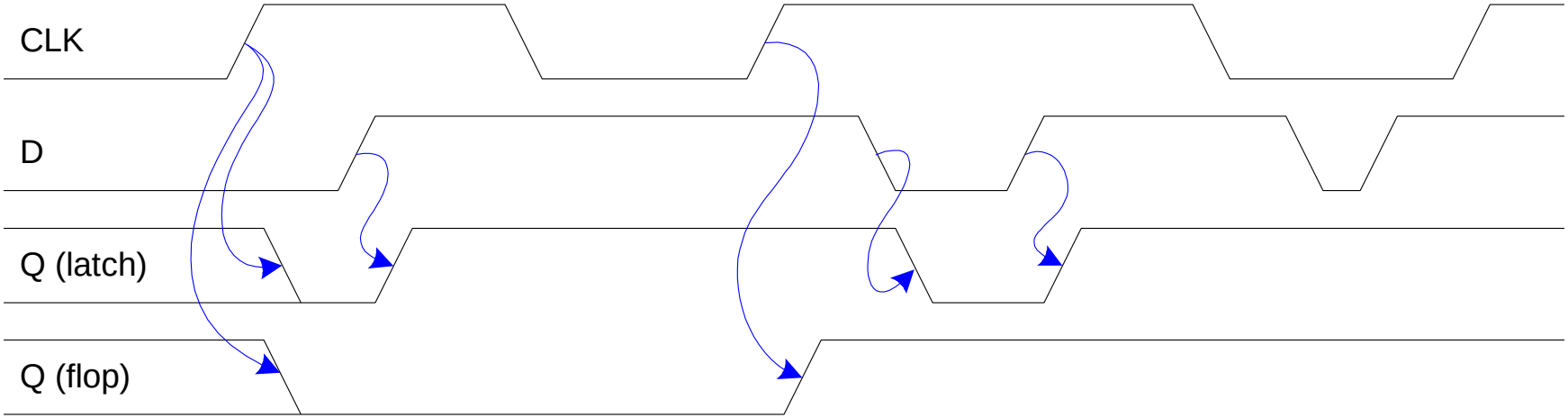
- **Critical (Long) Path:** $t_{pd} = 2t_{pd\_\mathrm{AND}} + t_{pd\_\mathrm{OR}}$

  Short Path: $t_{cd} = t_{cd\_\mathrm{AND}}$


Truth table ← Given Min/Max terms
K-map (minimized SOP)
Logic gate implementation
Timing diagram with logic gates delay consideration
Investigation of Timing hazards
Observation of the hazard (glitches from the osc)
Fixing Hazards

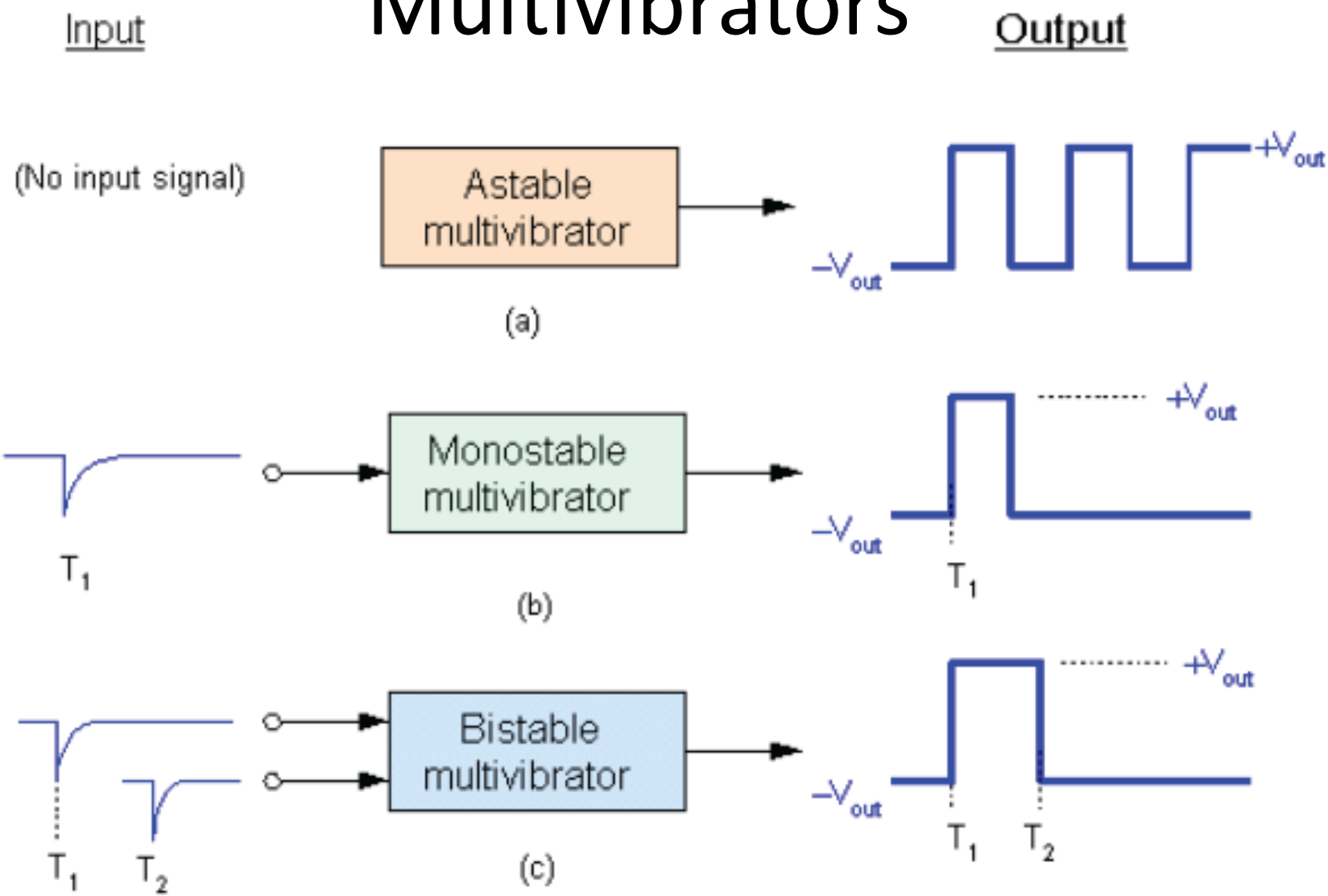
D Latch vs. D Flip-Flop



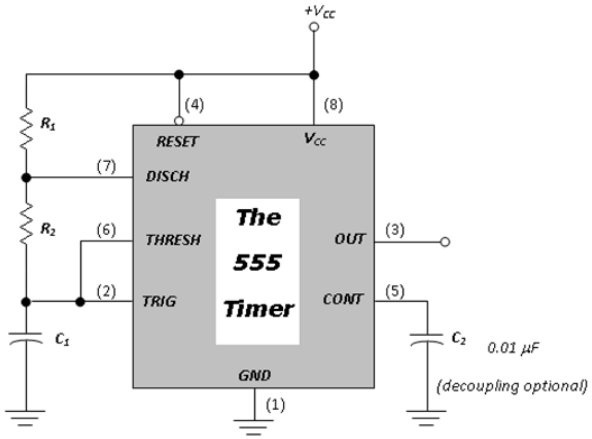


Level sensitive?
Edge sensitive?

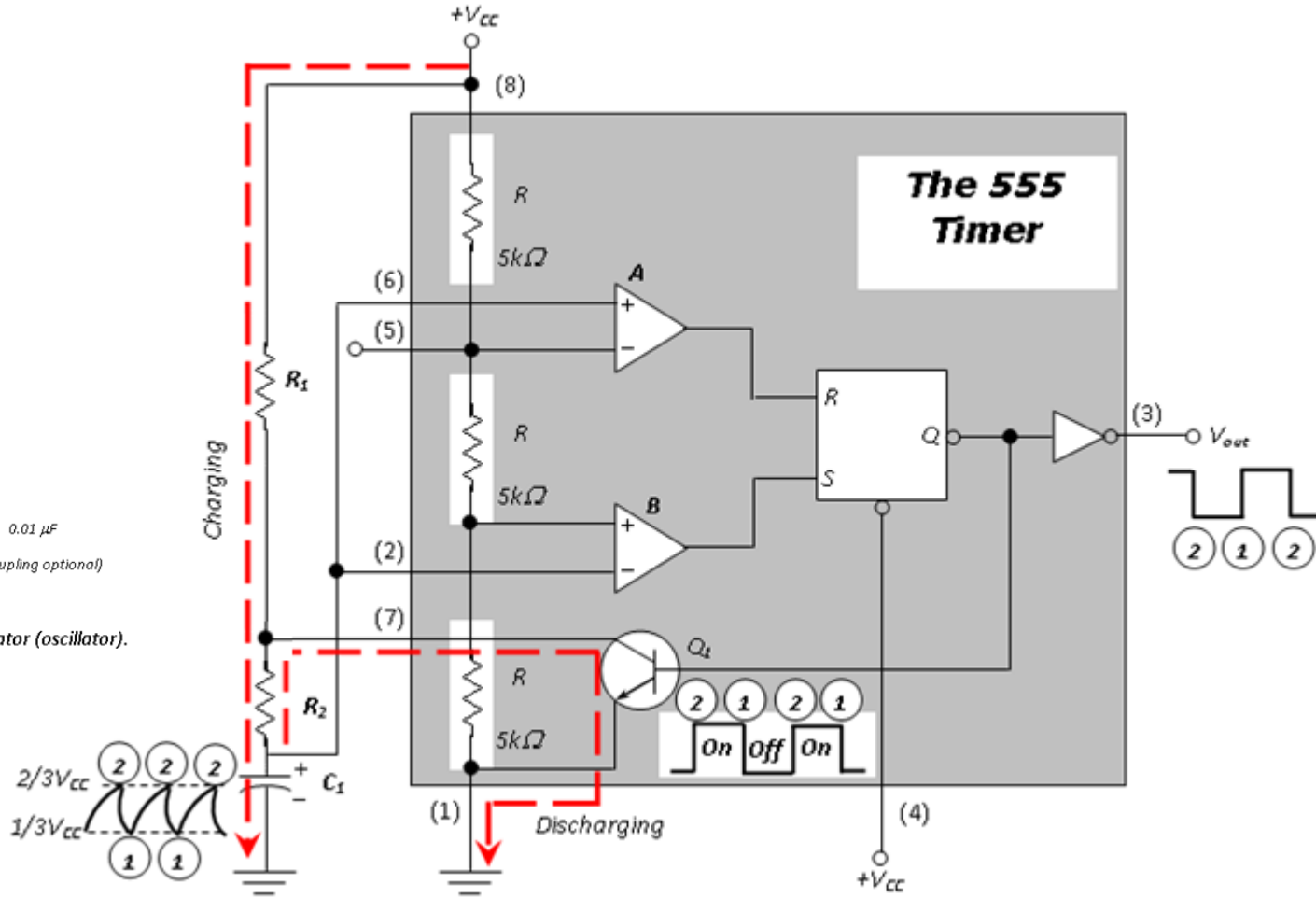
Multivibrators



555 timer



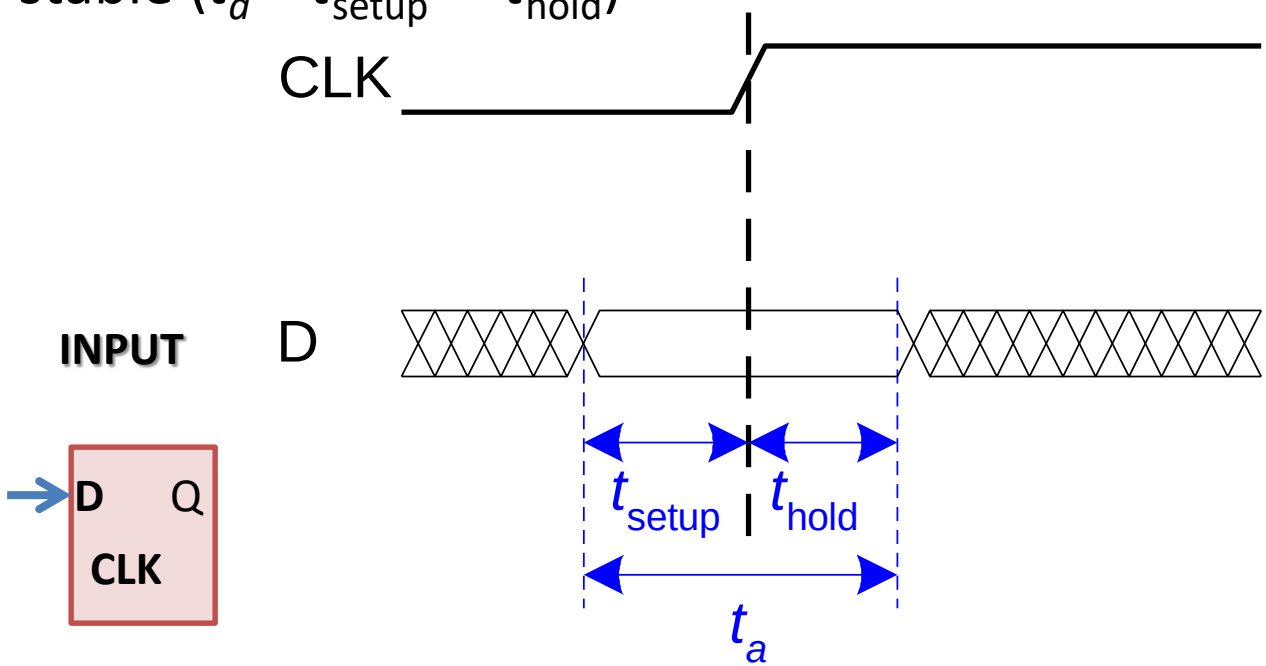
The 555 timer connected as an astable multivibrator (oscillator).



Looks very complicated!

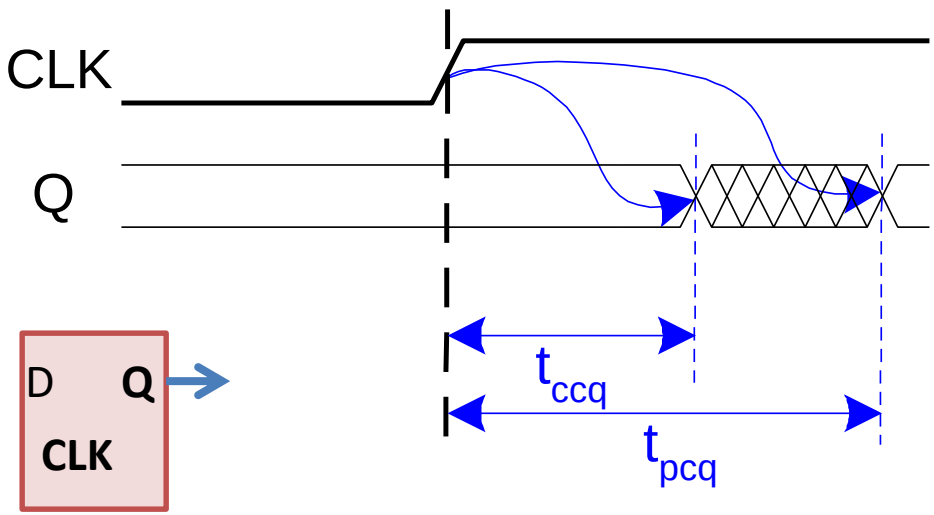
Input Timing

- **Setup time:** t_{setup} = time *before* clock edge data. It must be stable (i.e. not changing, guaranteed to be stable)
- **Hold time:** t_{hold} = time *after* clock edge data. It must be stable (guaranteed to be stable)
- Aperture time: t_a = time *around* clock edge data. It must be stable ($t_a = t_{setup} + t_{hold}$)



Output Timing

- **Propagation delay, clk to Q:** $\boldsymbol{t_{pcq}}$ = time after clock edge that the output <u>*Q* is guaranteed to be stable</u> (i.e., to stop changing, maximum t) ←→ t_{pd} in Comb. logics
- **Contamination delay, clk to Q:** $\boldsymbol{t_{ccq}}$ = time after clock edge that *Q* might be unstable (i.e., start changing, minimum t) ←→ t_{cd} in Comb. logics

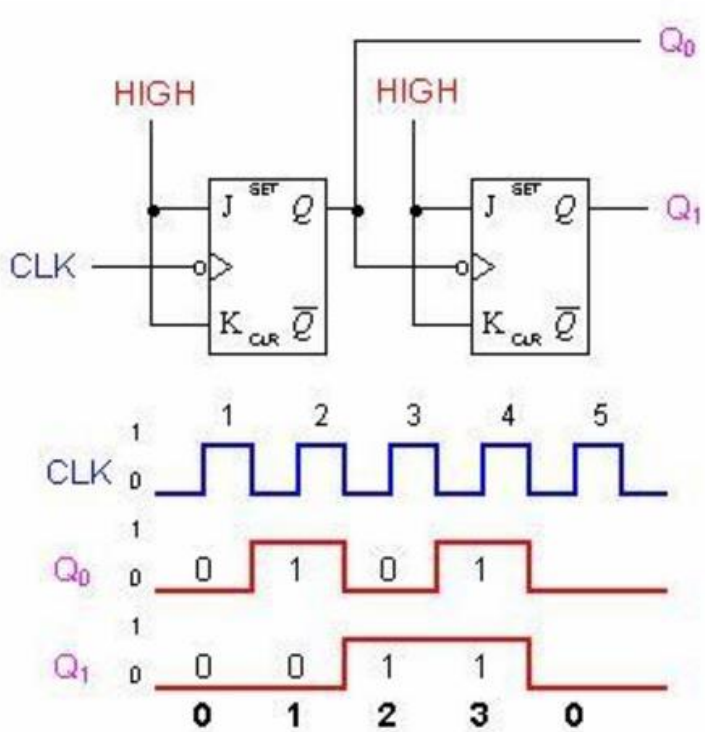


Asynchronous counters

-Ripple Counters-

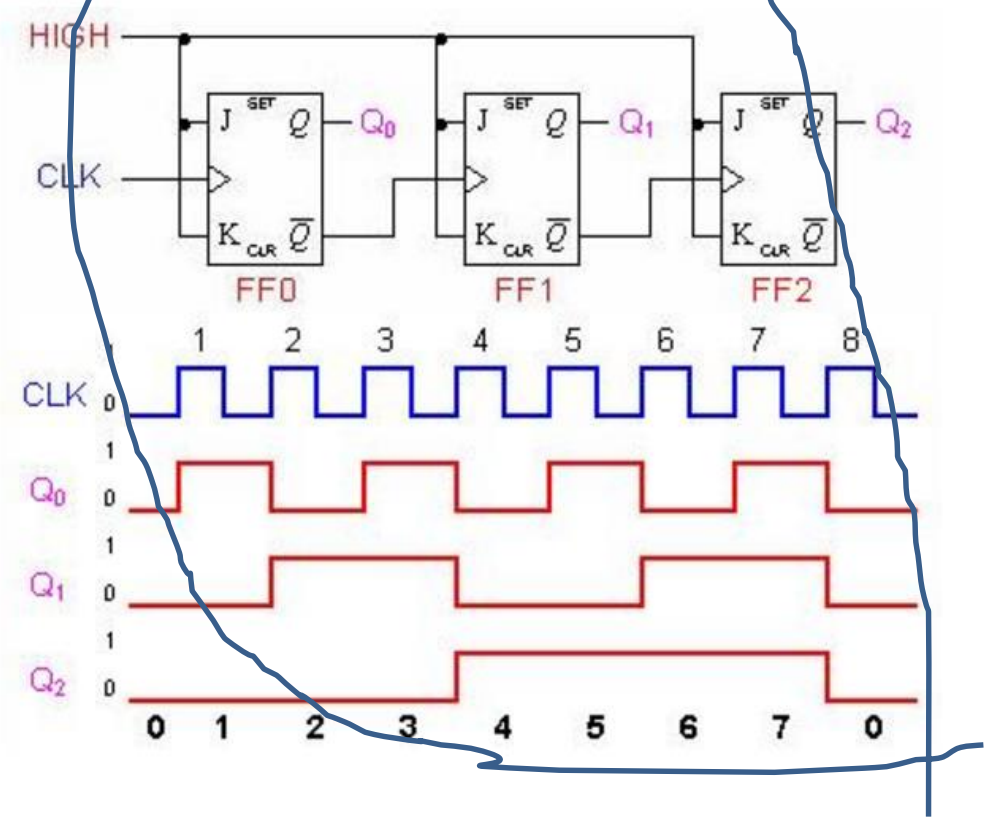
An asynchronous “up” counter: tell the difference

Negative-edge triggered flip-flops, connecting the clock inputs to the Q outputs of the preceding flip-flops.



A two-bit asynchronous counter

Positive-edge triggered flip-flops
Connecting the clock inputs to the Q’ outputs of the preceding flip-flops.

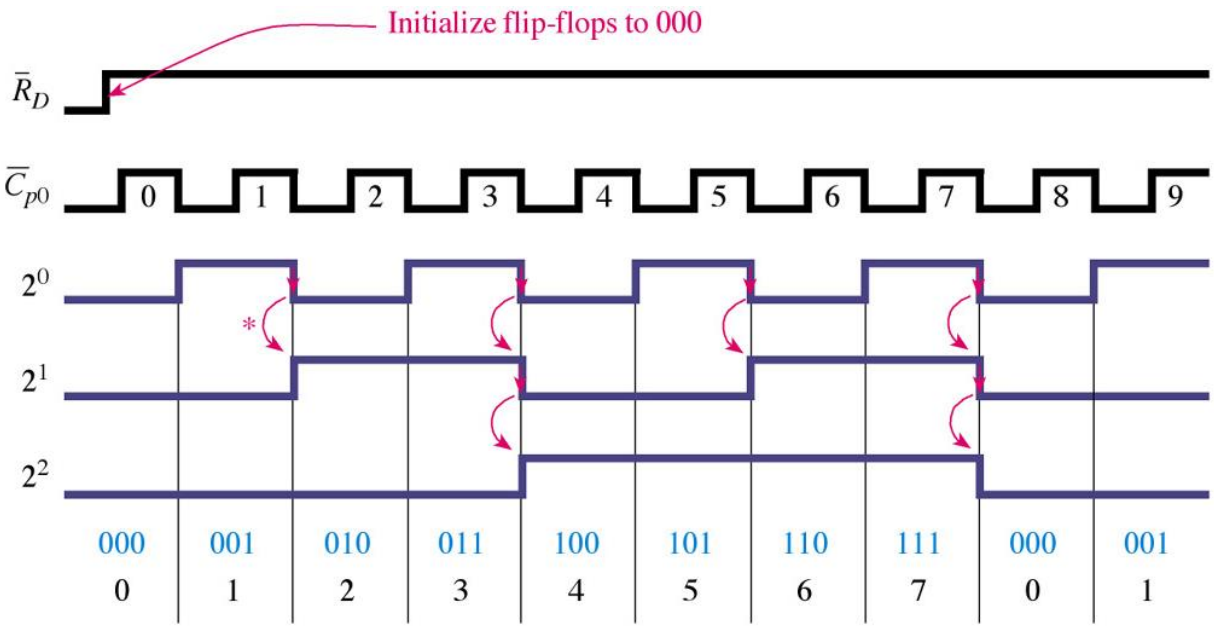


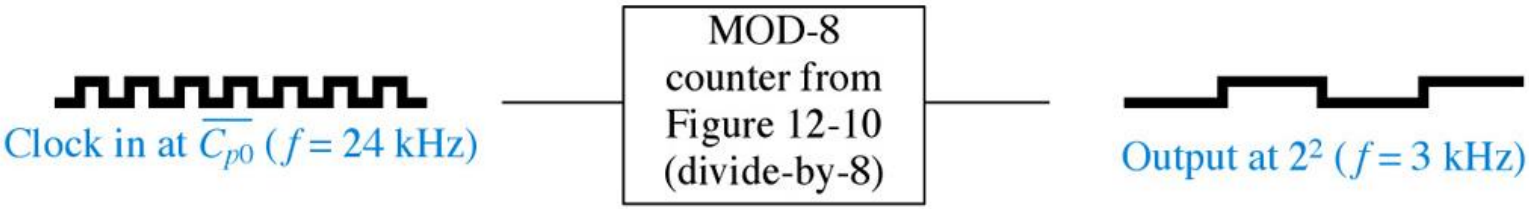
A three-bit asynchronous binary counter

https://www.ee.usyd.edu.au/tutorials/digital_tutorial/part2/counter02.html

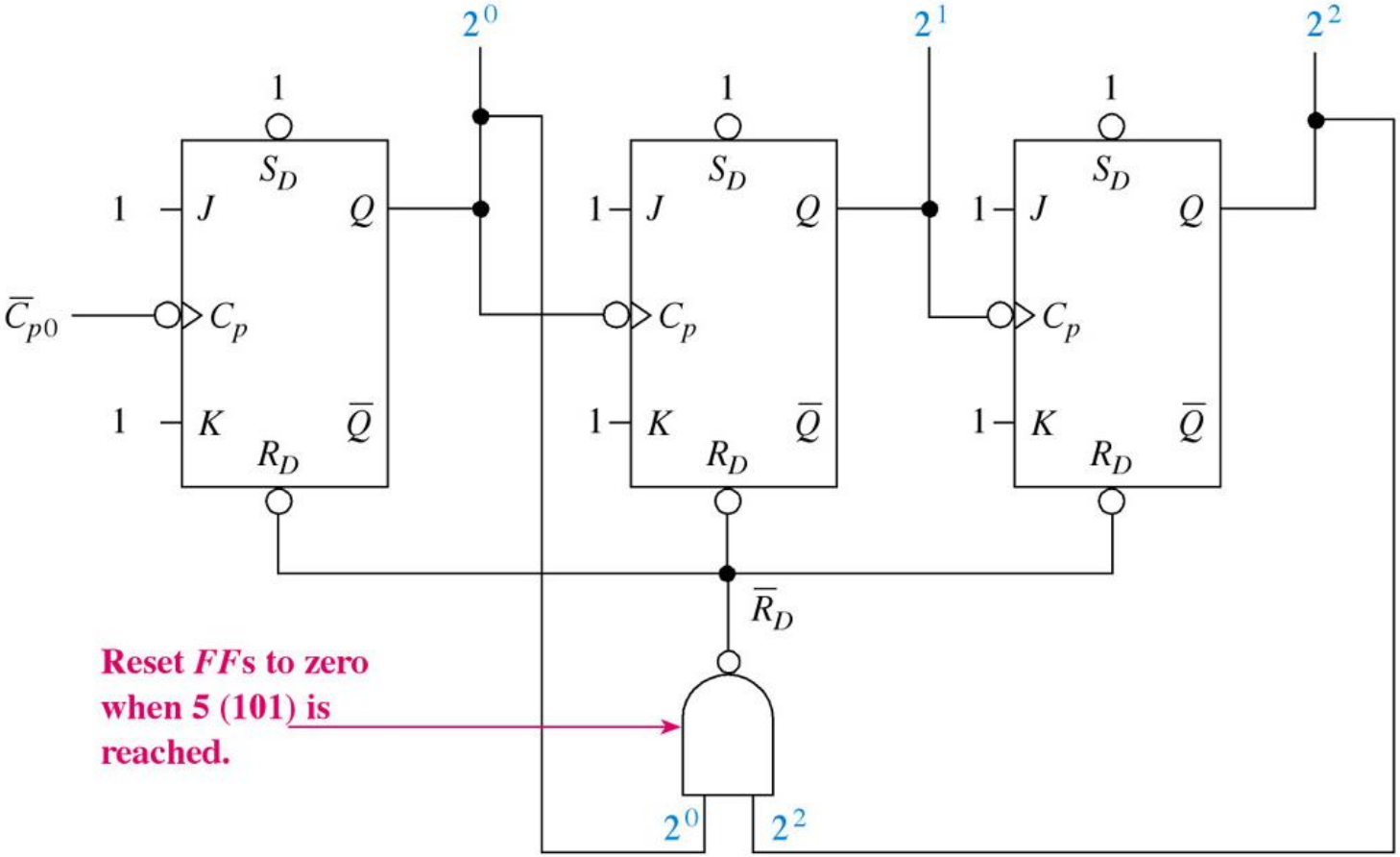
Divide-by-N Counters

- Reduce the frequency of periodic waveforms





- Divide-by-5 (MOD5) Counter



Synchronous counter

Synchronous counter implementations steps:

Steps

I. Decide the number of FFs (based on the states) and a kind of FF
   - i.e., 3 bits, JK FF

II. Excitation table of FF (relates Q_t and Q_{t+1})
   - Next slide

III. State diagram and circuit excitation table

IV. Obtain simplified equations using K map

V. Draw the logic diagram

V. Draw a diagram

$J_2=Q_1Q_0$

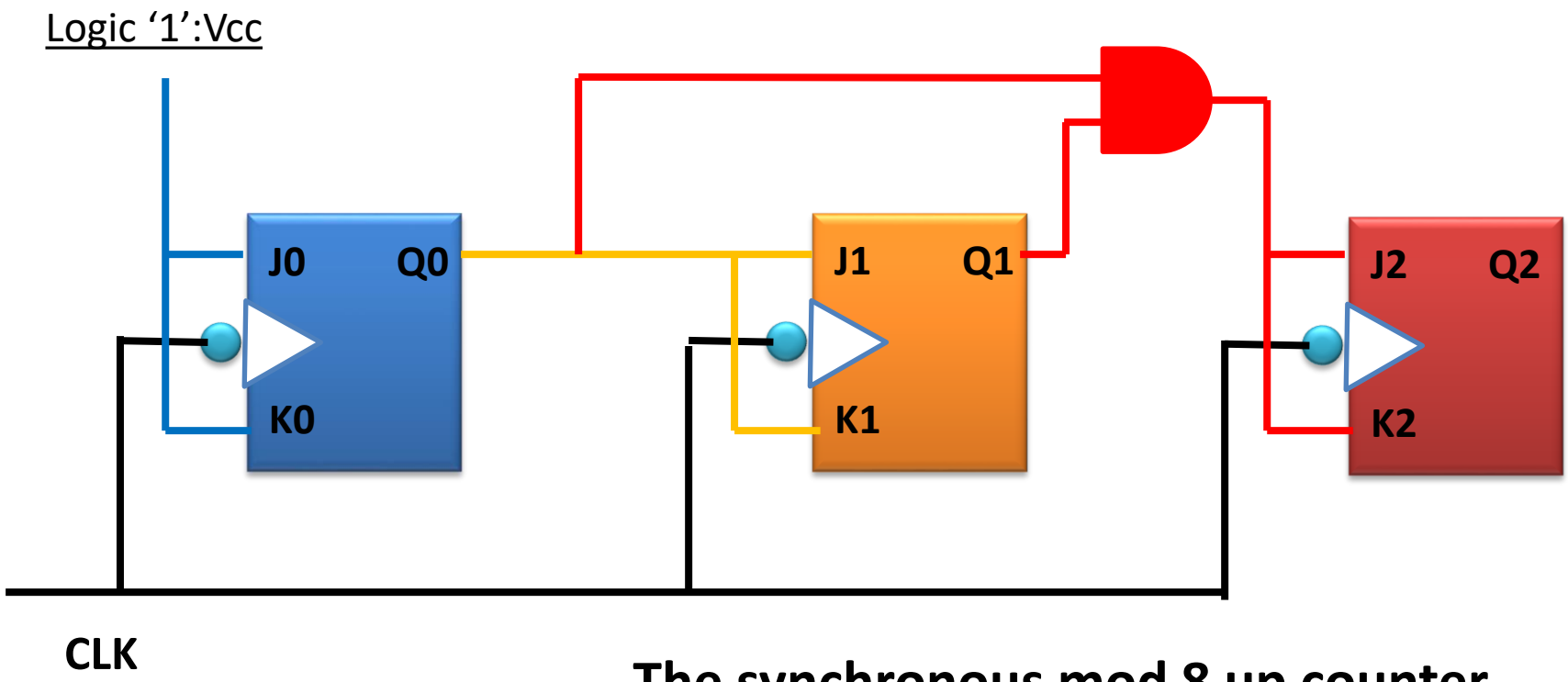
$J_1=Q_0$

$J_0=1$

$K_2=Q_1Q_0$

$K_1=Q_0$

$K_0=1$

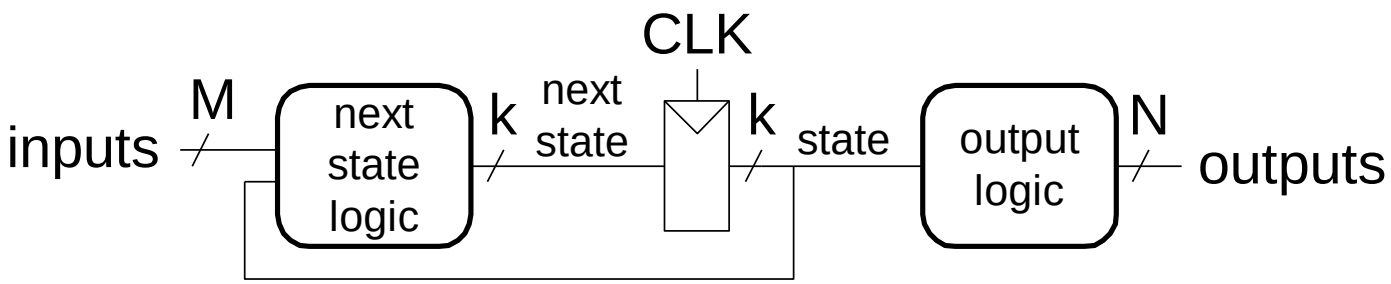


The synchronous mod 8 up counter

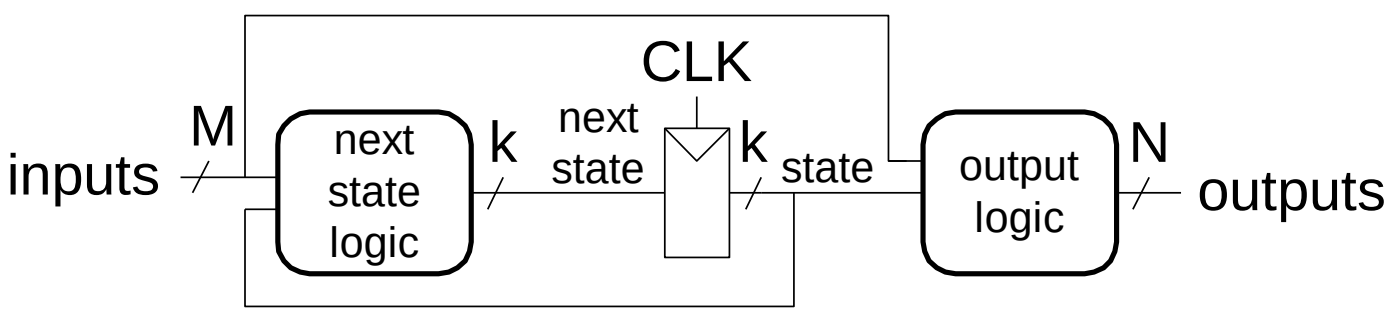
Day 15-16: State machine && Quiz 4

- **Next state** determined by current state and inputs
- Two types of finite state machines differ in **output logic**:
  - **Moore FSM:** outputs depend only on **current state**
  - **Mealy FSM:** outputs depend on **current state** *and* **inputs**

Moore FSM

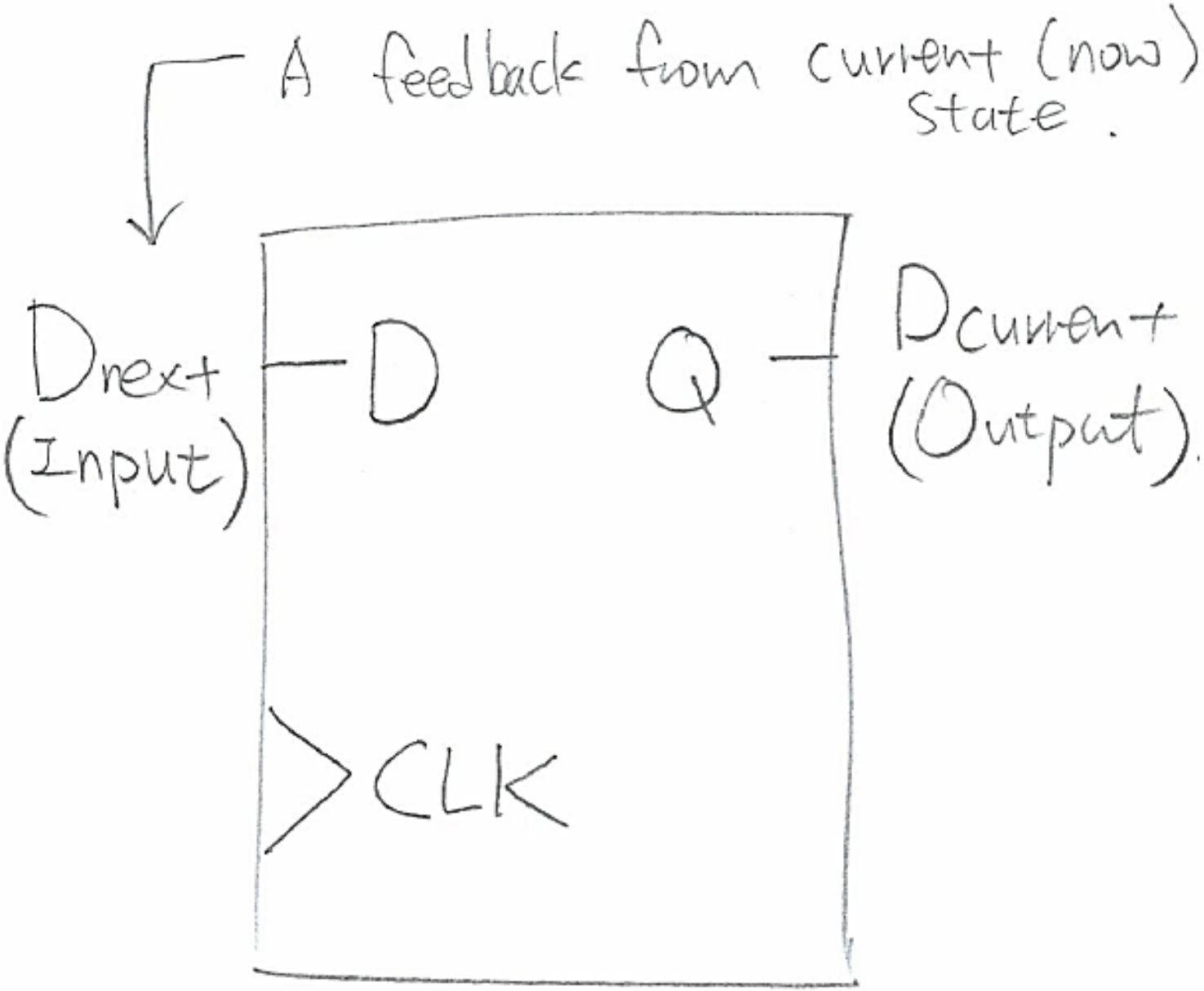


Mealy FSM

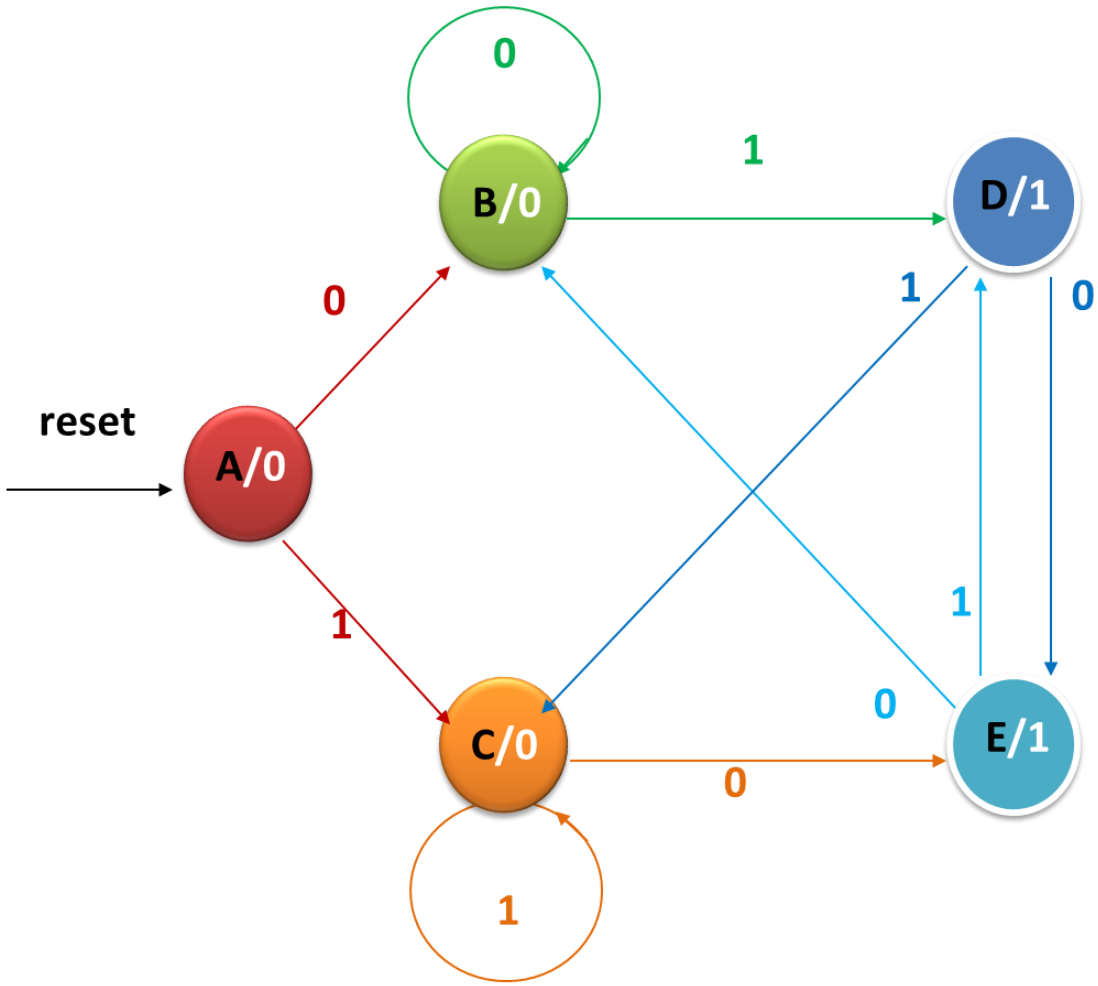


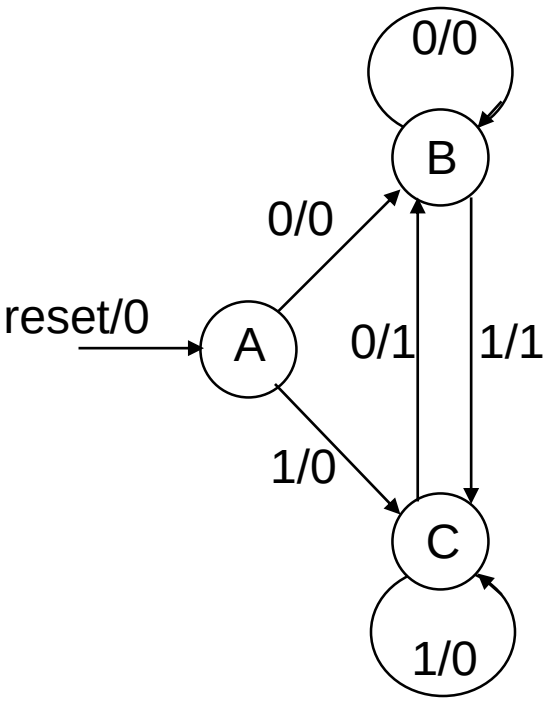
A general procedure of finite state machine implementation (6 steps)

1) Defining Inputs/Outputs from given problems
2) State Transition Diagram
3) Encoding (both states/inputs/outputs)
4) State Transition Table (both without and with encoding)
5) Next State Logic Boolean (with K-maps or Boolean simplification)
6) Output Logic Boolean (with K-map (or Boolean simplification) and Boolean expression)


A feedback from current (now) state.
Dnext
(Input)
D
Q
Dcurrent
(Output).
CLK

Moore vs. Mealy





3 bits vs. 2 bits

The End of CET 141