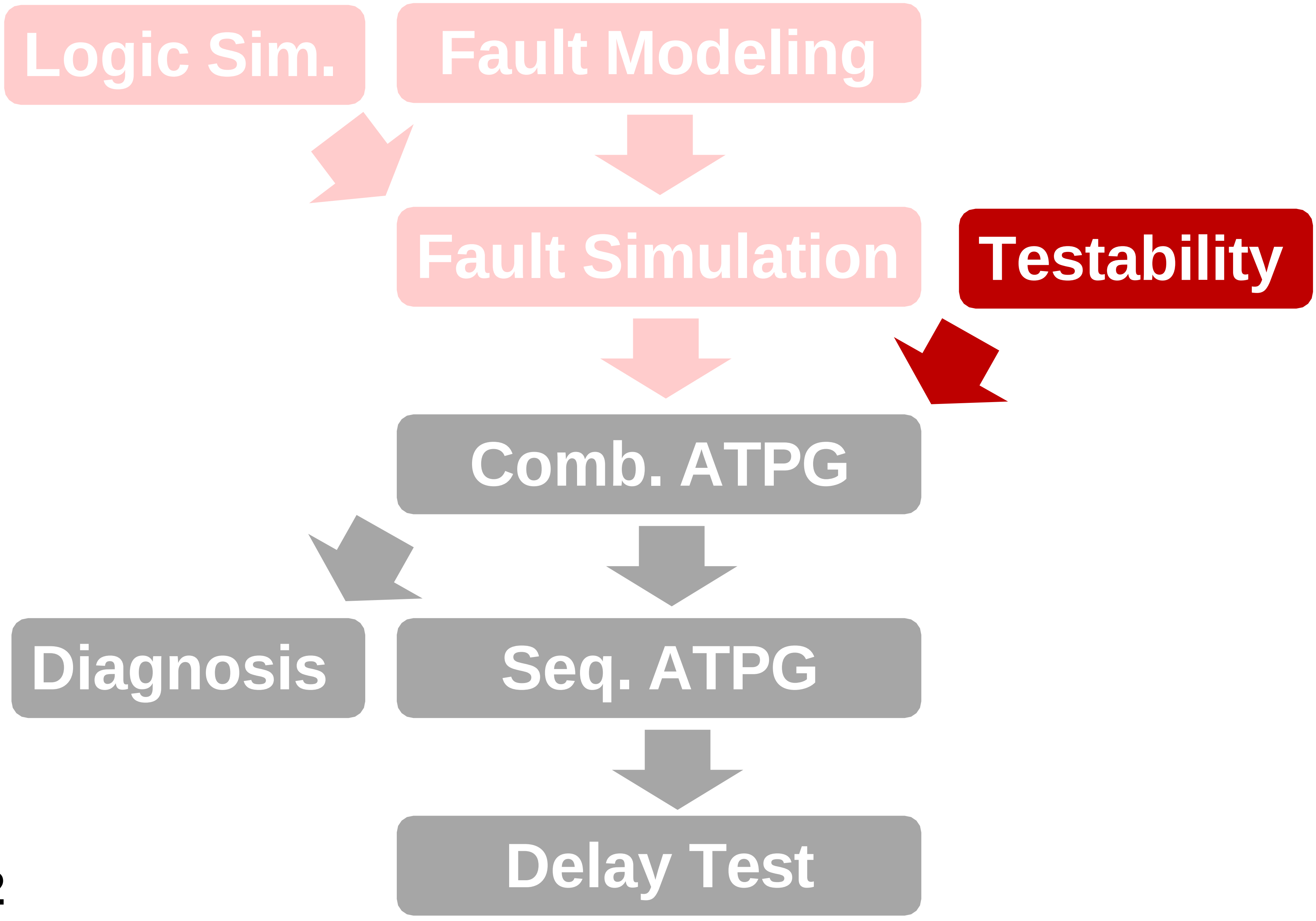
Testability Measures

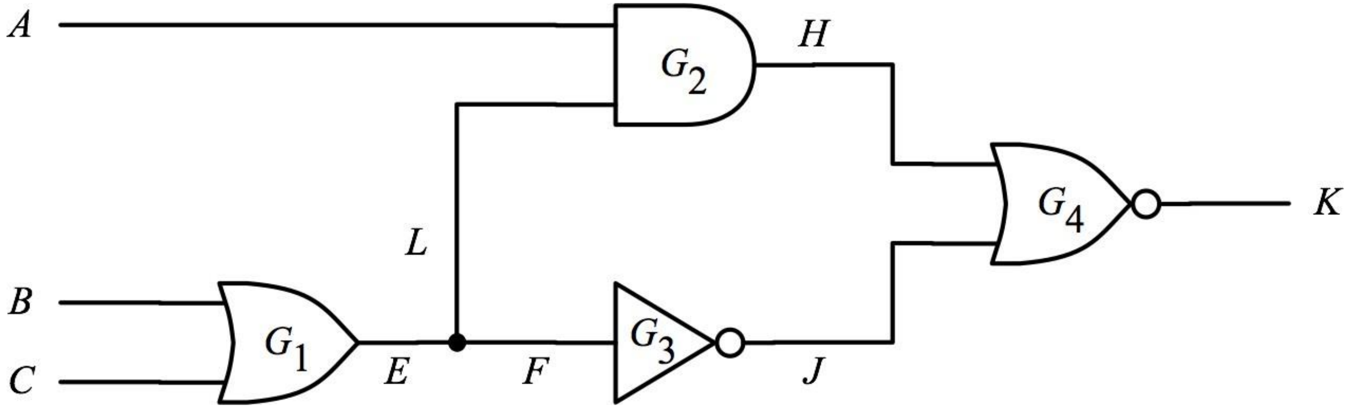
Dr. Bharat Garg
Assistant Professor,
DECE, TIET, Patiala

Course Roadmap (EDA Topics)



Motivating Problem

- **You report fault coverage of test set to your manager. Your manager is not very happy about FC number. He asked you: which faults are so difficult to detect?**



Why Am I Learning This?

- Testability measure helps to
  - Make smart decision in ATPG
  - More testable designs

"Measurement is the first step that leads to control and eventually to improvement."

(H. James Harrington)

Testability Measures

- What is testability measure?
  - Metric to measure degree of difficulty to test a circuit

- Two important components:
  - *Controllability*
    - degree of difficulty to control a logic signal to 0 or 1
  - *Observability*
    - degree of difficulty to observe the logic value of a signal

Testability Measures Controllability and Observability

Testability Analysis

- What is ***Testability Analysis***?
  - Calculate testability measures for a given circuit
- Why testability analysis?
  - 1. Help ATPG make smart decision
  - 2. Insert DFT circuits to improve controllability and observability
- When testability analysis?
  - In *preprocess* stage of ATPG or DFT insertion
- Requirements of testability analysis
  - Should run very fast
  - Sometimes , accuracy is not very important

Categories of Testability Analysis

- 1. ***Topology-based*** **analysis**
  - **Only analyzes structure of circuit. No test vectors are given.**
  - **Example: SCOAP**
- 2. ***Probability-based*** **analysis**
  - **Uses *signal probability* to estimate the testability**
  - **Example: COP**
- 3. ***High-level*** **analysis**
  - **Performs testability analysis before synthesis**
  - **Example: RTL testability analysis**

- 4. ***Simulation-base*** **analysis (not in lecture)**
  - **Apply input patterns,**
  - **Perform simulation and estimate testability**

SCOAP [Goldstein 1979]

- *Sandia Controllability Observability Analysis Program**
- SCOAP computes 6 numbers for each node *N*

*Sandia is name of research lab.

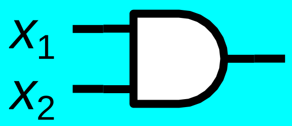
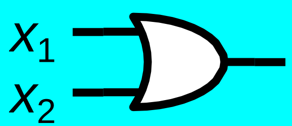
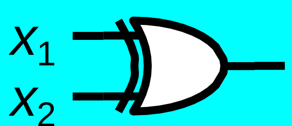
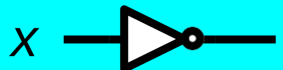
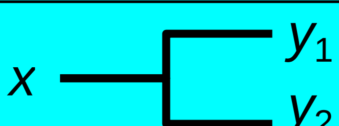
	0-controllability	1-controllability	Observability
Combinational	$CC^0(N)$	$CC^1(N)$	$CO(N)$
Sequential	$SC^0(N)$	$SC^1(N)$	$SO(N)$

Combinational Controllability

- ***$CC^0(N)$, $CC^1(N)$***
  - **minimum number of combinational PI assignments and logic levels required to control a 0 or a 1 on node *N***
  - ***Smaller* number, *easier* to control**
- **How to calculate CC?**
  - **CC(PI) = 1.**
  - **From PI to PO. Add 1 to account for logic level**
- **Gate propagation rules:**
  - **If only one input controls gate output:**
    - **CC(gate_output) = min { CC(gate_input) } + 1**
  - **If all inputs needed to set gate output:**
    - **CC(gate_output) = Σ CC(gate_input) + 1**
  - **CC(branches)=CC(stem)**

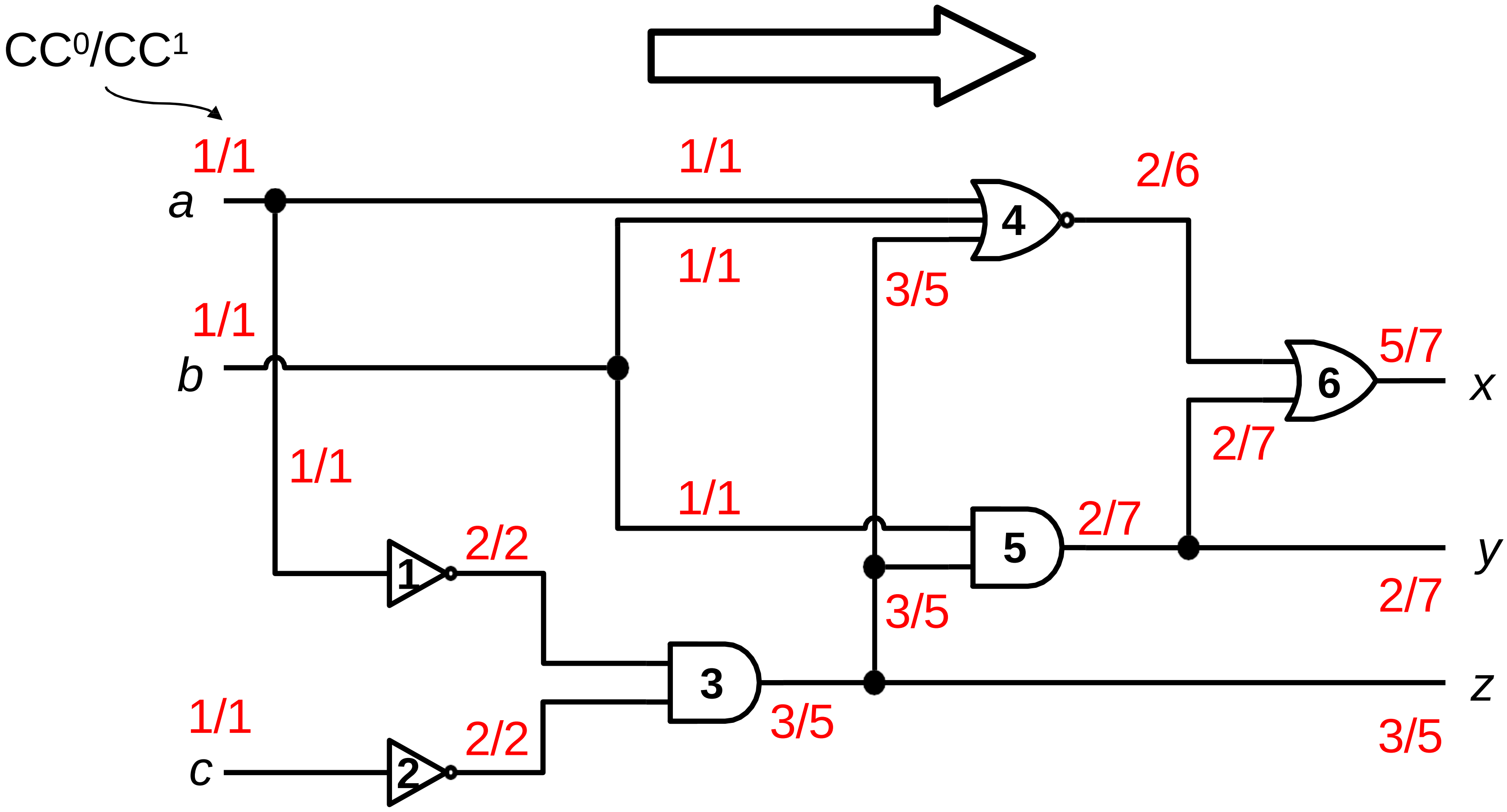
Smaller CC is Better

$CC^0(N)$ & $CC^1(N)$

	$CC^0(y)$	$CC^1(y)$
Primary inputs	1	1
	$\min[CC^0(x_1), CC^0(x_2)] + 1$	$CC^1(x_1) + CC^1(x_2) + 1$
	$CC^0(x_1) + CC^0(x_2) + 1$	$\min[CC^1(x_1), CC^1(x_2)] + 1$
	$\min[CC^0(x_1) + CC^0(x_2),$ $CC^1(x_1) + CC^1(x_2)] + 1$	$\min[CC^0(x_1) + CC^1(x_2),$ $CC^1(x_1) + CC^0(x_2)] + 1$
	$CC^1(x) + 1$	$CC^0(x) + 1$
	$CC^0(y_1) = CC^0(y_2) = CC^0(x)$	$CC^1(y_1) = CC^1(y_2) = CC^1(x)$

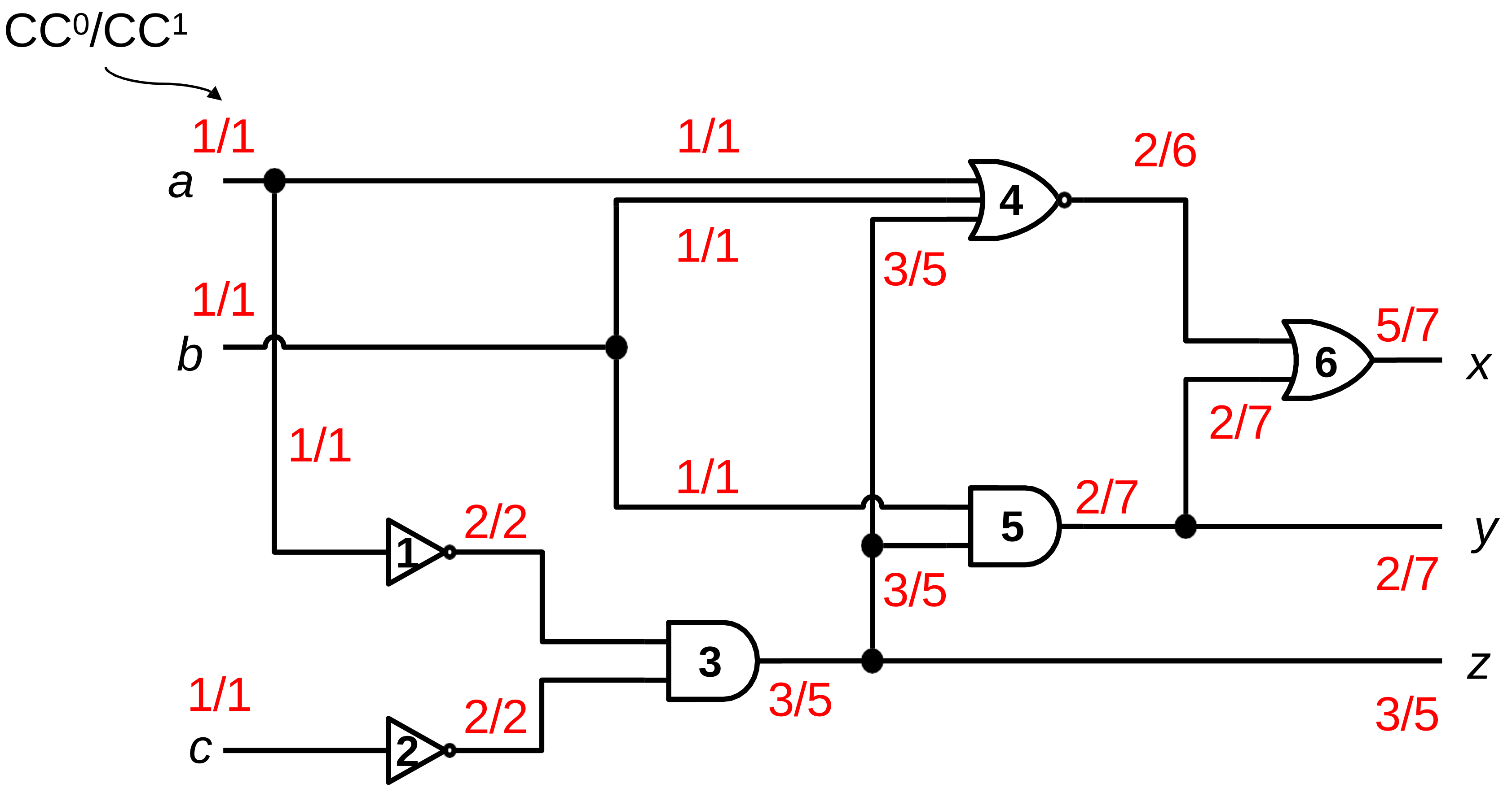
An Example – Controllability

- **Forward: From PI to PO**



Quiz

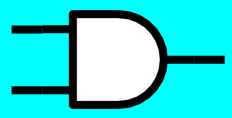
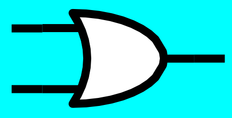
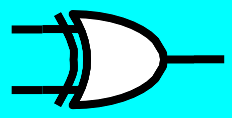
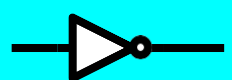
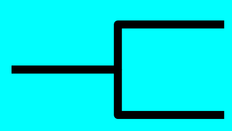
Q: how to control *y* to 0? How to control *y* to 1?



Combinational Observability

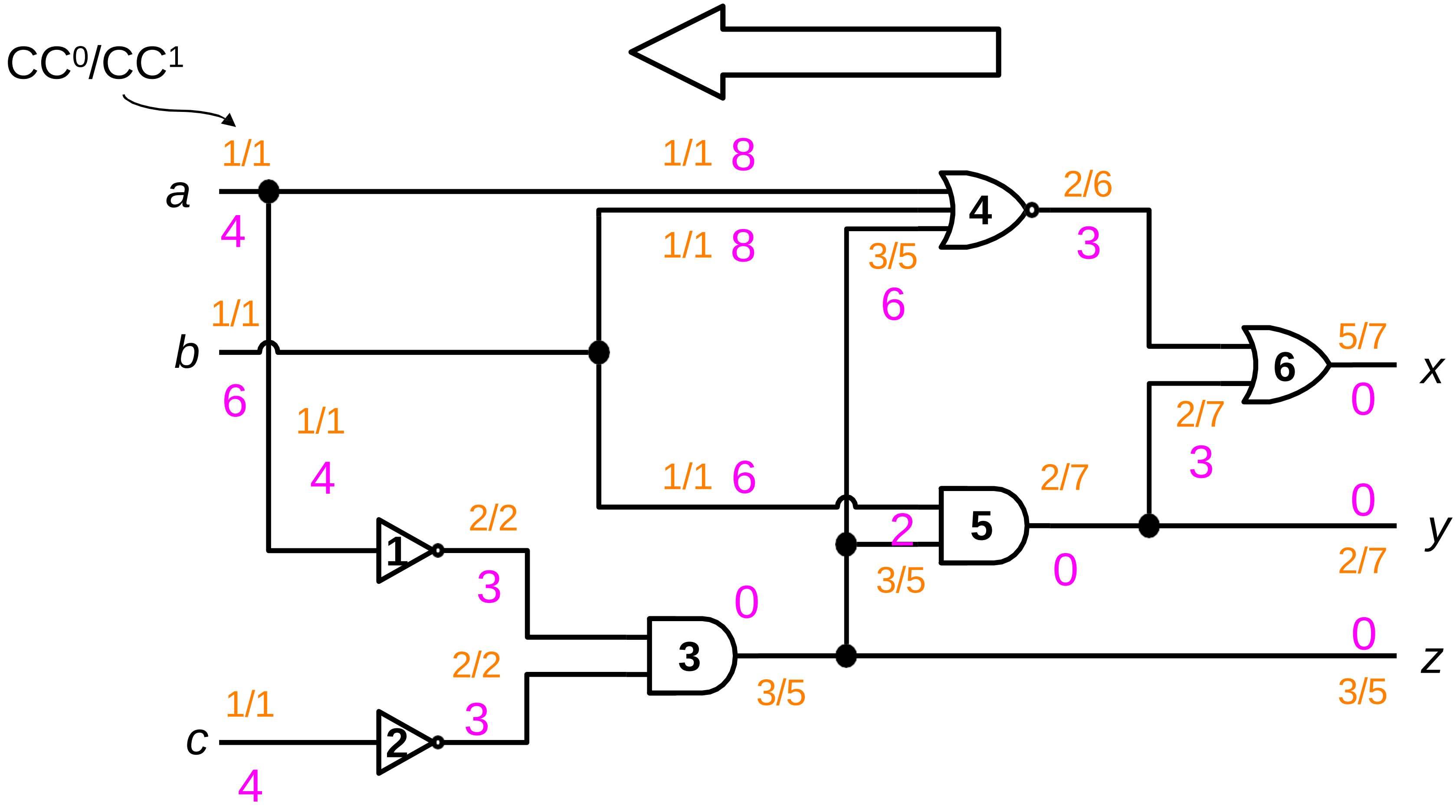
- ***CO(N)***
  - minimum number of combinational PI assignments and logic levels required to propagate logical value on node *N* to PO
  - ***Smaller*** **number,** ***easier*** **to observe**
- **How to calculate CO(N)?**
  - **CO(PO)=0**
  - **From POs to PIs, add 1 to account for logic level**
- **CO(gate_input)=** Σ
  - **(1) CO(gate_output)**
  - **(2) CC(setting all other inputs to non-controlling value)**
  - **(3) + 1 for logic level**
- **How about fanout stem? Assume they are independent**
  - **CO(stem) = min { CO(branches) }**

CO(N)

	$CO(x_1)$
Primary outputs	**0**
x_1 x_2  y	$CO(y) + CC^1(x_2) + 1$
x_1 x_2  y	$CO(y) + CC^0(x_2) + 1$
x_1 x_2  y	$CO(y) + \min[CC^0(x_2), CC^1(x_2)] + 1$
x_1  y	$CO(y) + 1$
x_1  y_1 y_2	$\min[CO(y_1), CO(y_2)]$

An Example – Observability

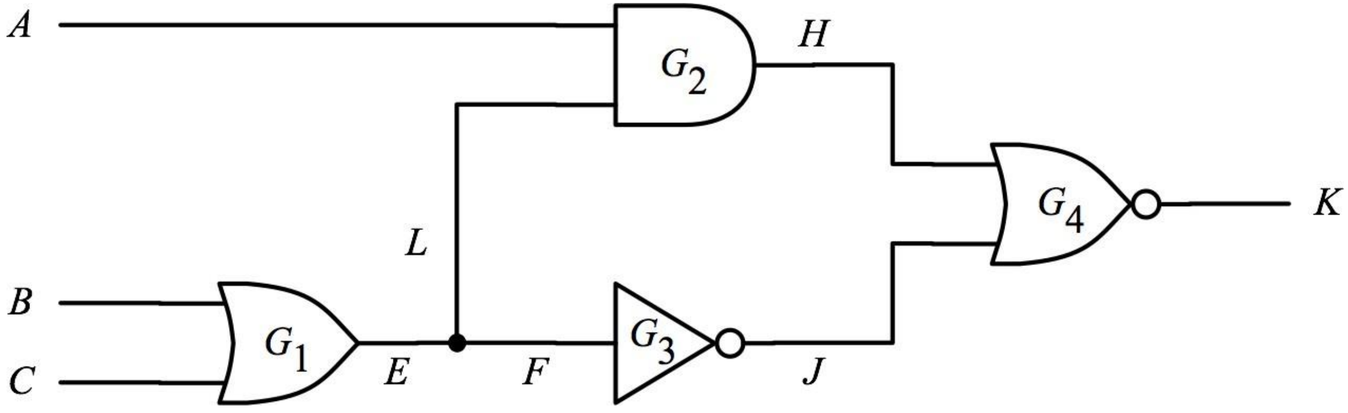
- **Backward: From PO to PI**



Quiz

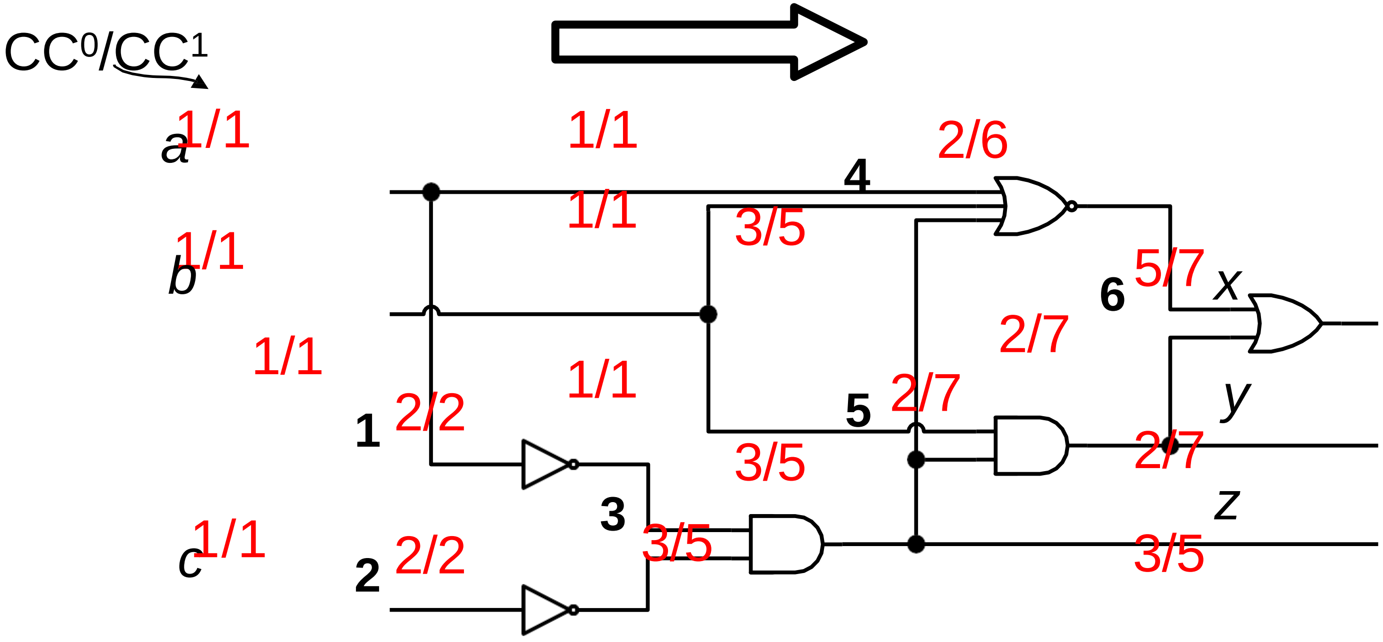
Q: Please analyze combinational SCOAP.
A:

	A	B	C	E	F	L	H	J	K
CC^0									
CC^1									
CO									



FFT

- **Q: Testability should be done very quickly. What is time complexity to calculate CC and CO?**

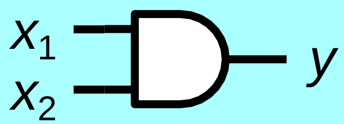
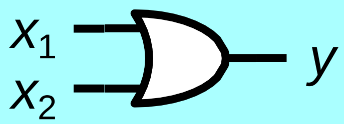
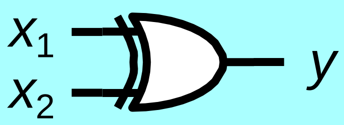
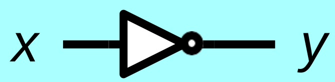
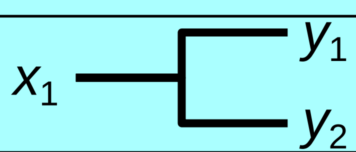


Sequential SCOAP Measures

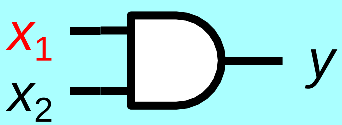
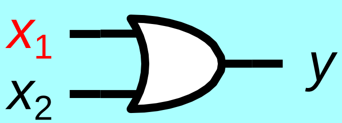
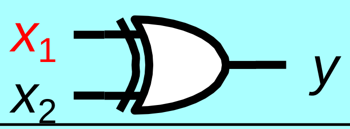
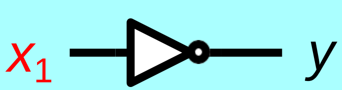
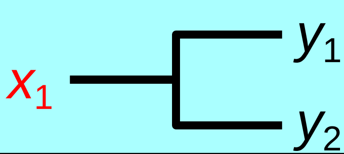
- ***Sequential controllability: $SC^0(N)$, $SC^1(N)$***
  - **Minimum number of FF assignments (number of clock cycles) required to control 0 or 1 on node *N***
  - **smaller number means easier to control**
- ***Sequential observability: SO(N)***
  - **Minimum number of FF assignments required to propagate logical value on node *N* to a primary output**
- **NOTE: assume no scan**
  - **Can only control PI, observe PO**
  - **Can NOT control FF, can NOT observe FF**

Sequential SCOAP Measures
# of Clock Cycles Needed

$SC^0(N)$ and $SC^1(N)$

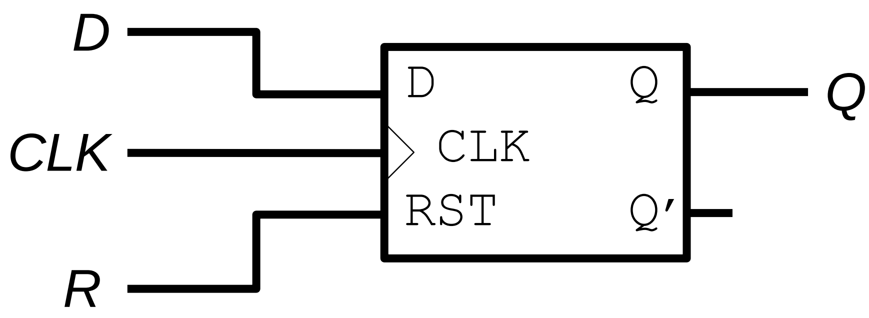
	$SC^0(y)$	$SC^1(y)$
Primary inputs	**0 (not 1)**	**0**
	$\min[SC^0(x_1), SC^0(x_2)]$ $\cancel{+1}$	$SC^1(x_1) + SC^1(x_2)$
	$SC^0(x_1) + SC^0(x_2)$	$\min[SC^1(x_1), SC^1(x_2)]$
	$\min[SC^0(x_1) + SC^0(x_2),$ $SC^1(x_1) + SC^1(x_2)]$	$\min[SC^0(x_1) + SC^1(x_2),$ $SC^1(x_1) + SC^0(x_2)]$
	$SC^1(x)$	$SC^0(x)$
	$SC^0(y_1) = SC^0(y_2) = SC^0(x_1)$	$SC^1(y_1) = SC^1(y_2) = SC^1(x_1)$

SO(N)

	$SO(x_1)$
Primary outputs	**0**
	$SO(y) + SC^1(x_2)$ ~~+1~~
	$SO(y) + SC^0(x_2)$
	$SO(y) + \min[SC^0(x_2), SC^1(x_2)]$
	$SO(y)$
	$\min[SO(y_1), SO(y_2)]$

Flip-Flop (Controllability)

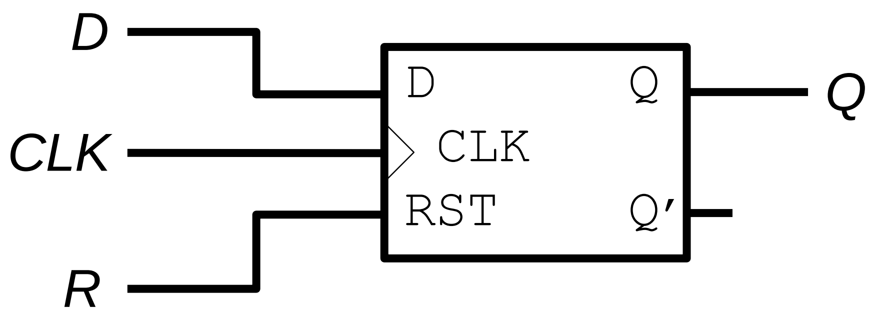
- **Positive edge triggered, asynchronous reset**



$$CC^1(Q) = CC^1(D) + CC^1(CLK) + CC^0(CLK) + CC^0(R)$$
$$SC^1(Q) = SC^1(D) + SC^1(CLK) + SC^0(CLK) + SC^0(R) + 1$$

$$CC^0(Q) = \min[CC^1(R),\ CC^0(D) + CC^1(CLK) + CC^0(CLK) + CC^0(R)]$$
$$SC^0(Q) = \min[SC^1(R),\ SC^0(D) + SC^1(CLK) + SC^0(CLK) + SC^0(R)] + 1$$

Flip-Flop (Observability)



$$CO(D) = CO(Q) + CC^1(CLK) + CC^0(CLK) + CC^0(R)$$

$$SO(D) = SO(Q) + SC^1(CLK) + SC^0(CLK) + SC^0(R) + 1$$

Seq. SCOAP Computation Alg.

- **Computation of SC, SO is similar to CC, CO**
  - **but require *iterations* for controllability to converge**

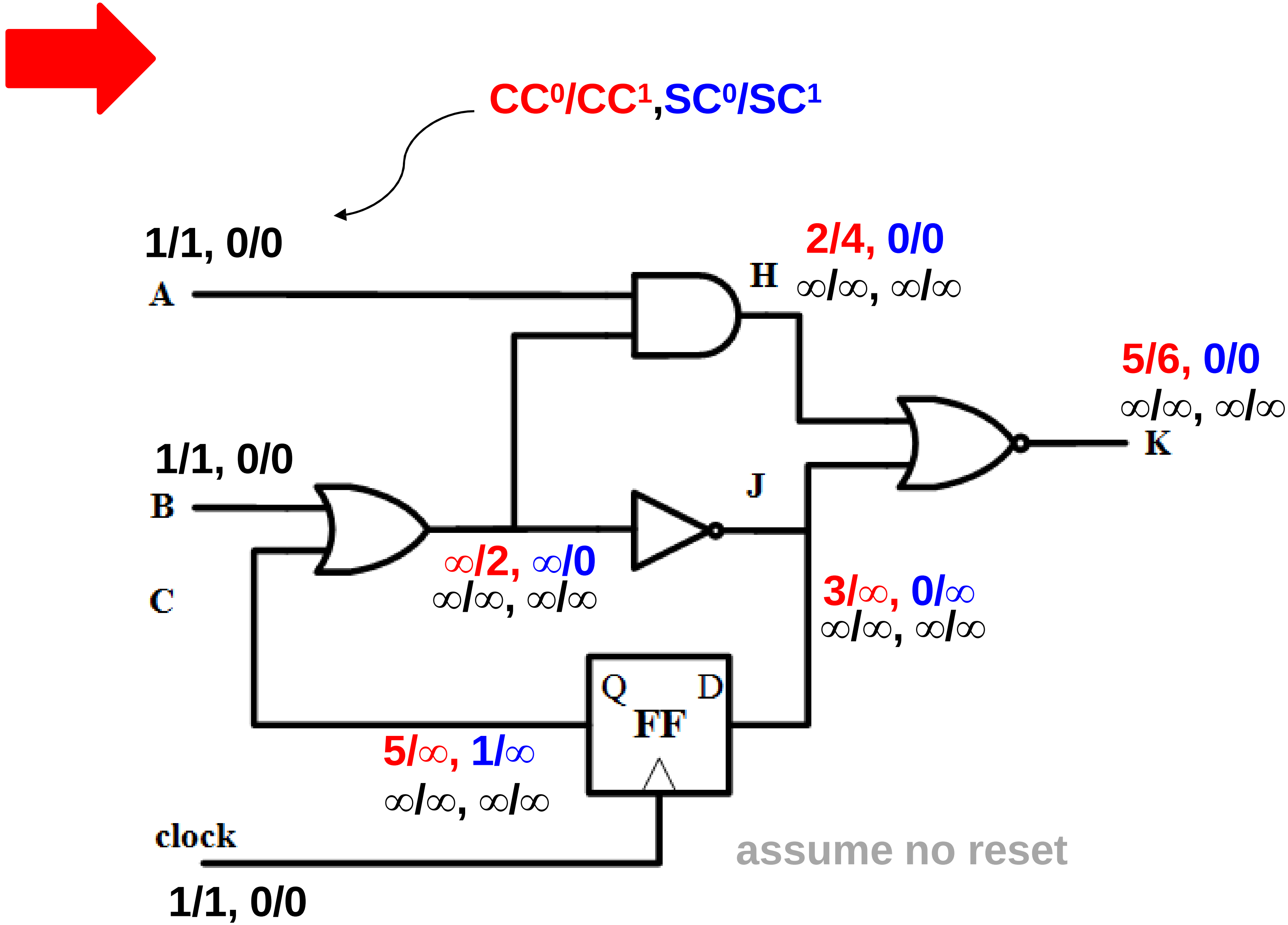
Controllability:

1. **For all PI's, set $CC^0 = CC^1 = 1$ and $SC^0 = SC^1 = 0$**
2. **For all other nodes, set $CC^0 = CC^1 = \infty$ and $SC^0 = SC^1 = \infty$**
3. **Propagate controllability from PI's to PO's**
   Iterate until numbers stabilize.

Observability:

1. **For all PO's, set CO = SO = 0**
2. **For all other nodes, set CO = SO = ∞**
3. **Propagate observability from PO's to PI's**
   (note: no iteration needed for CO/SO)

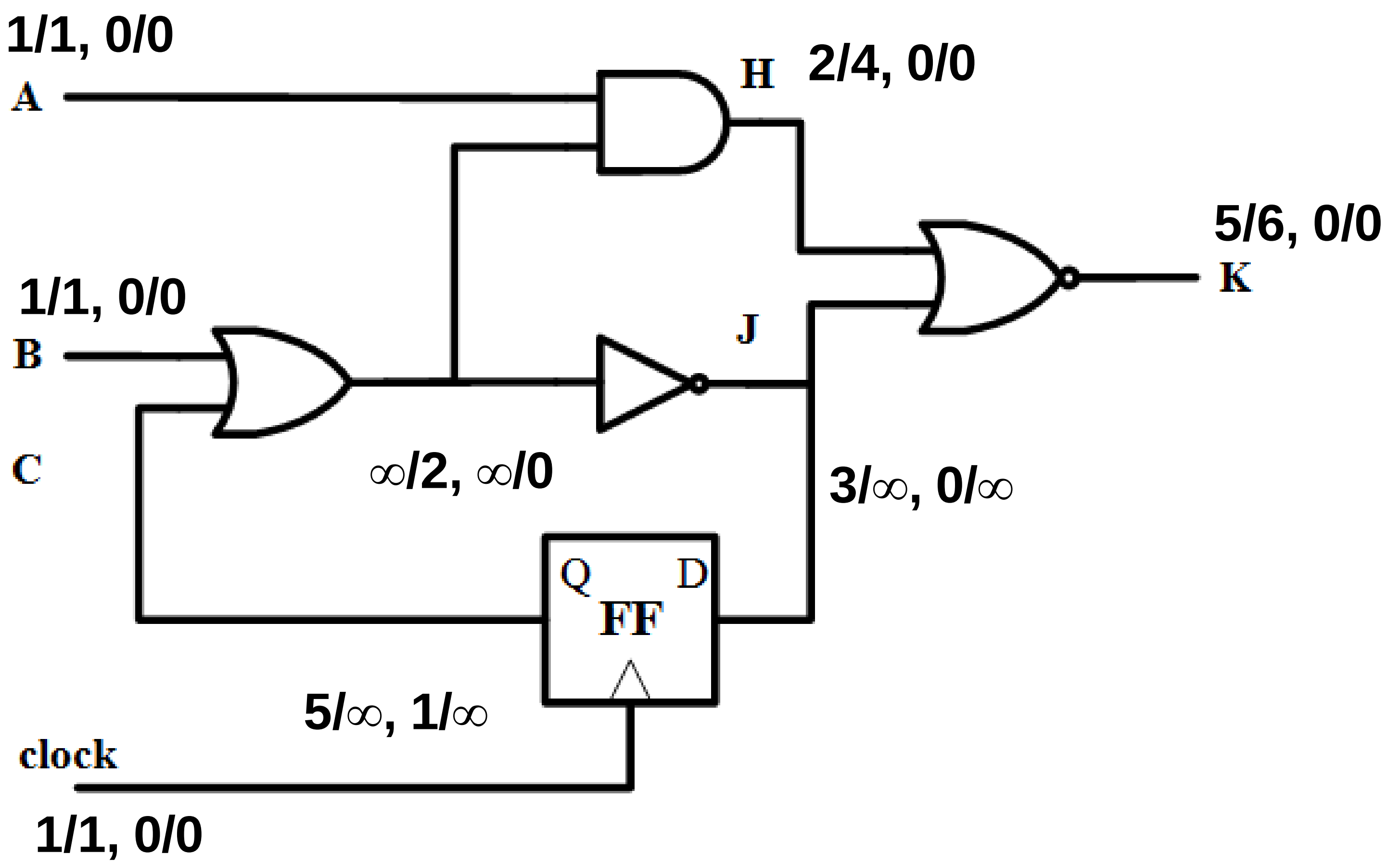
Controllability Computation - 1



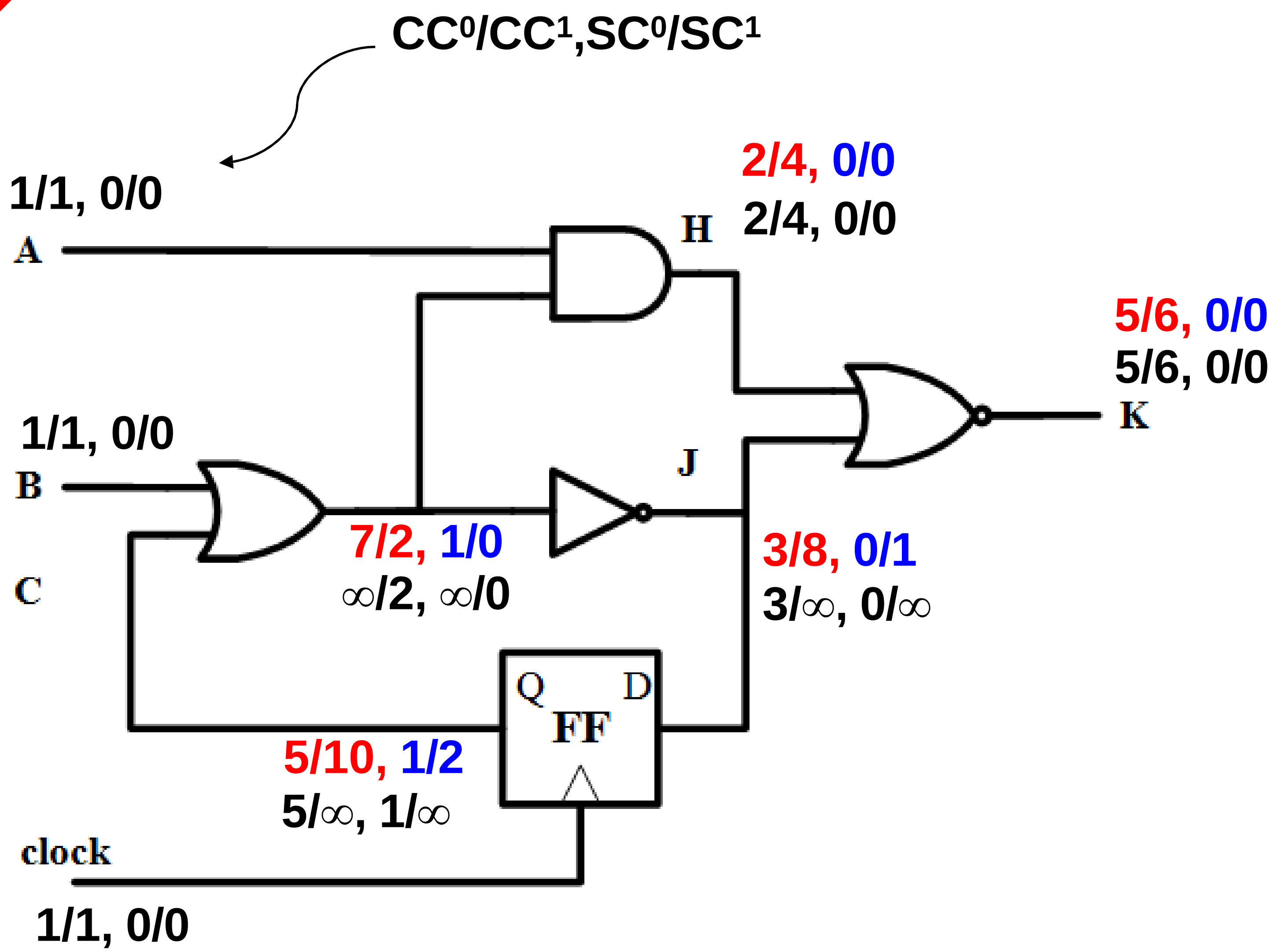
assume no reset

Quiz

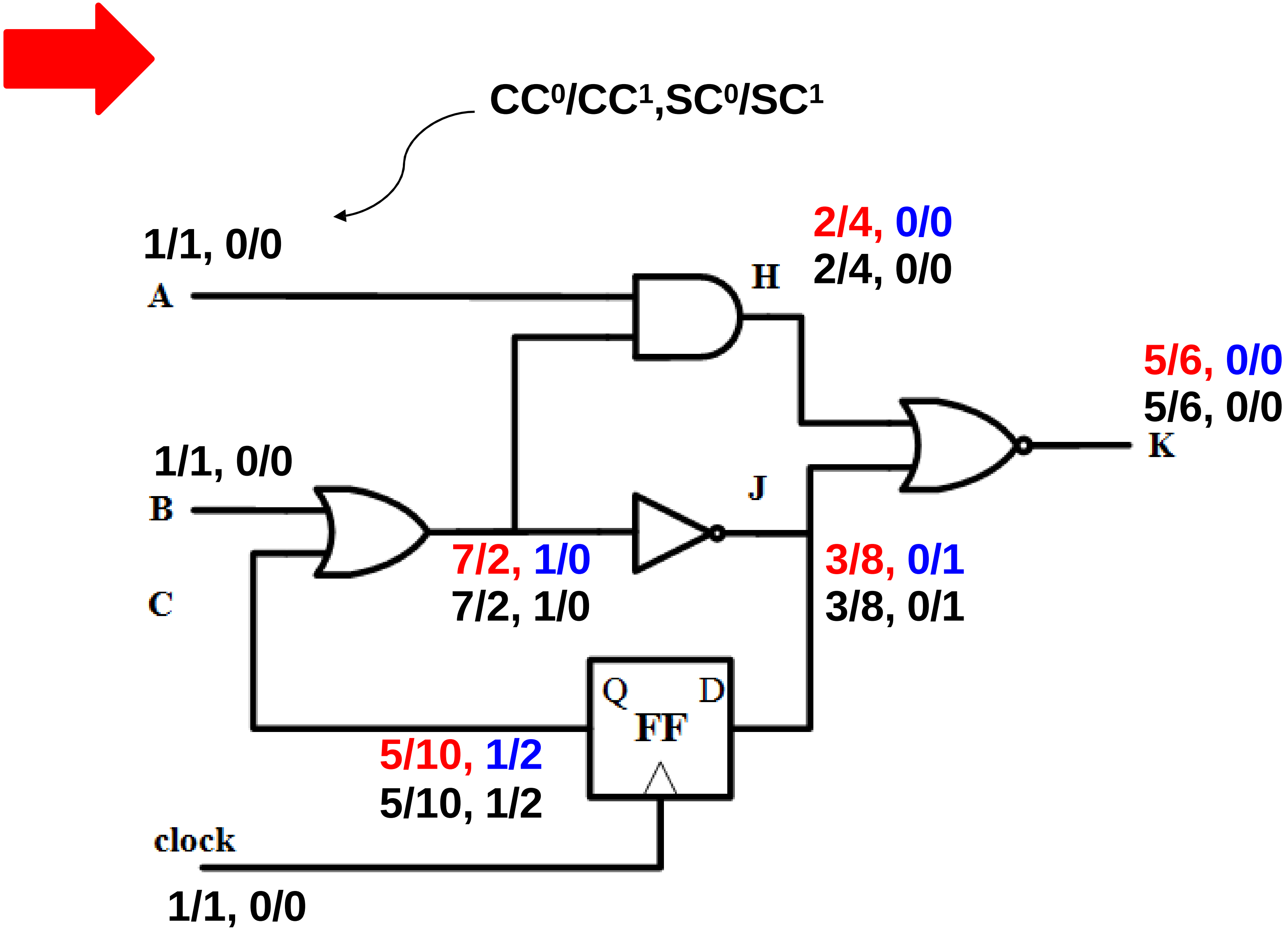
Q: Given numbers from the 1st iteration, please continue to calculate CC0/ CC1, SC0/SC1 in 2nd iteration.



Controllability Computation - 2

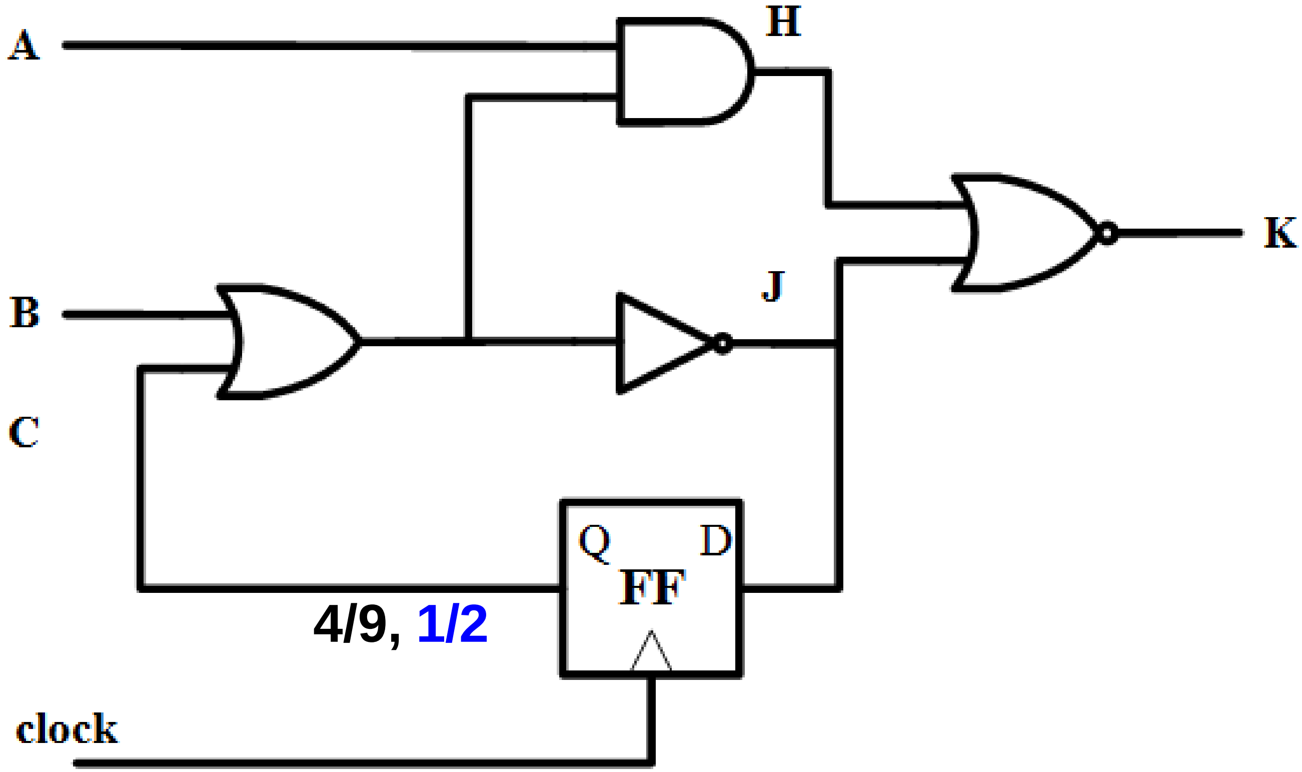


Controllability Computation - 3



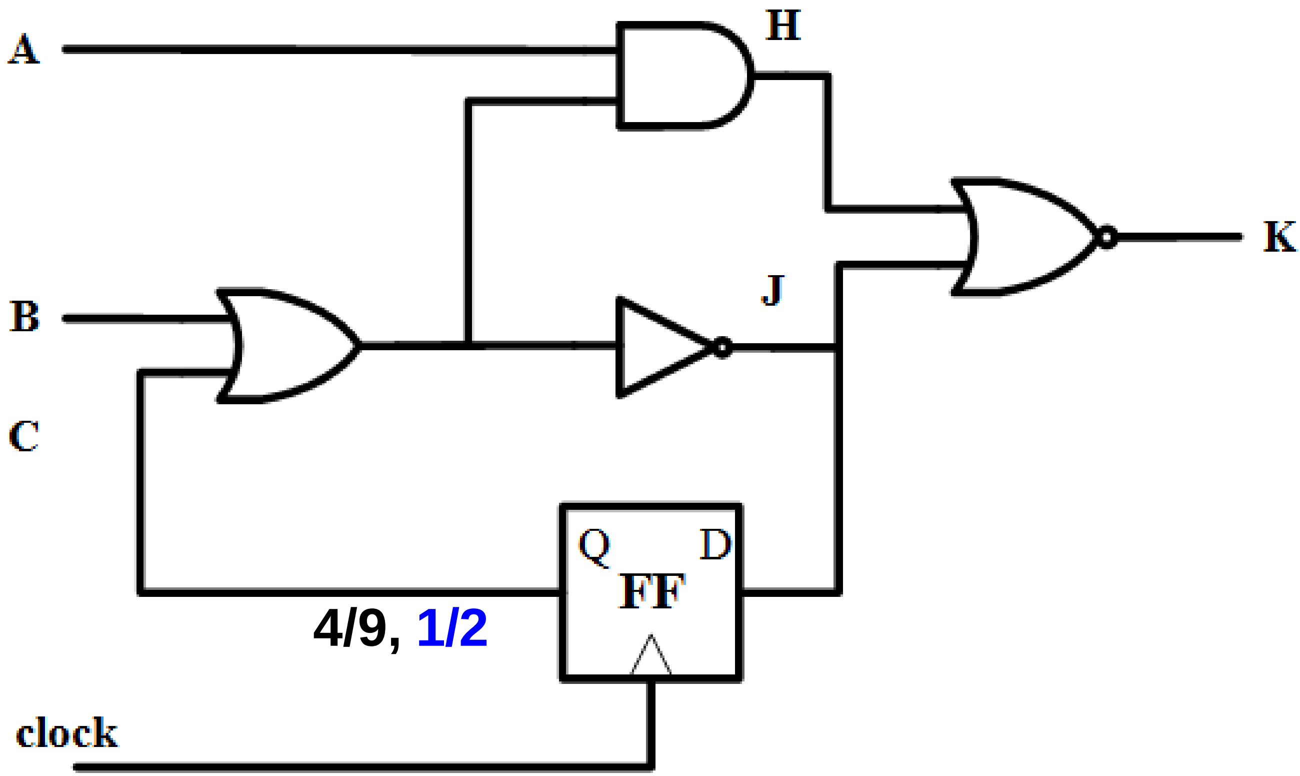
Quiz

Q1: Generate a sequence of test patterns to control *C* to 0?
Q2: Generate a sequence of test patterns to control *C* to 1?
(assume no scan. can only assign PI)

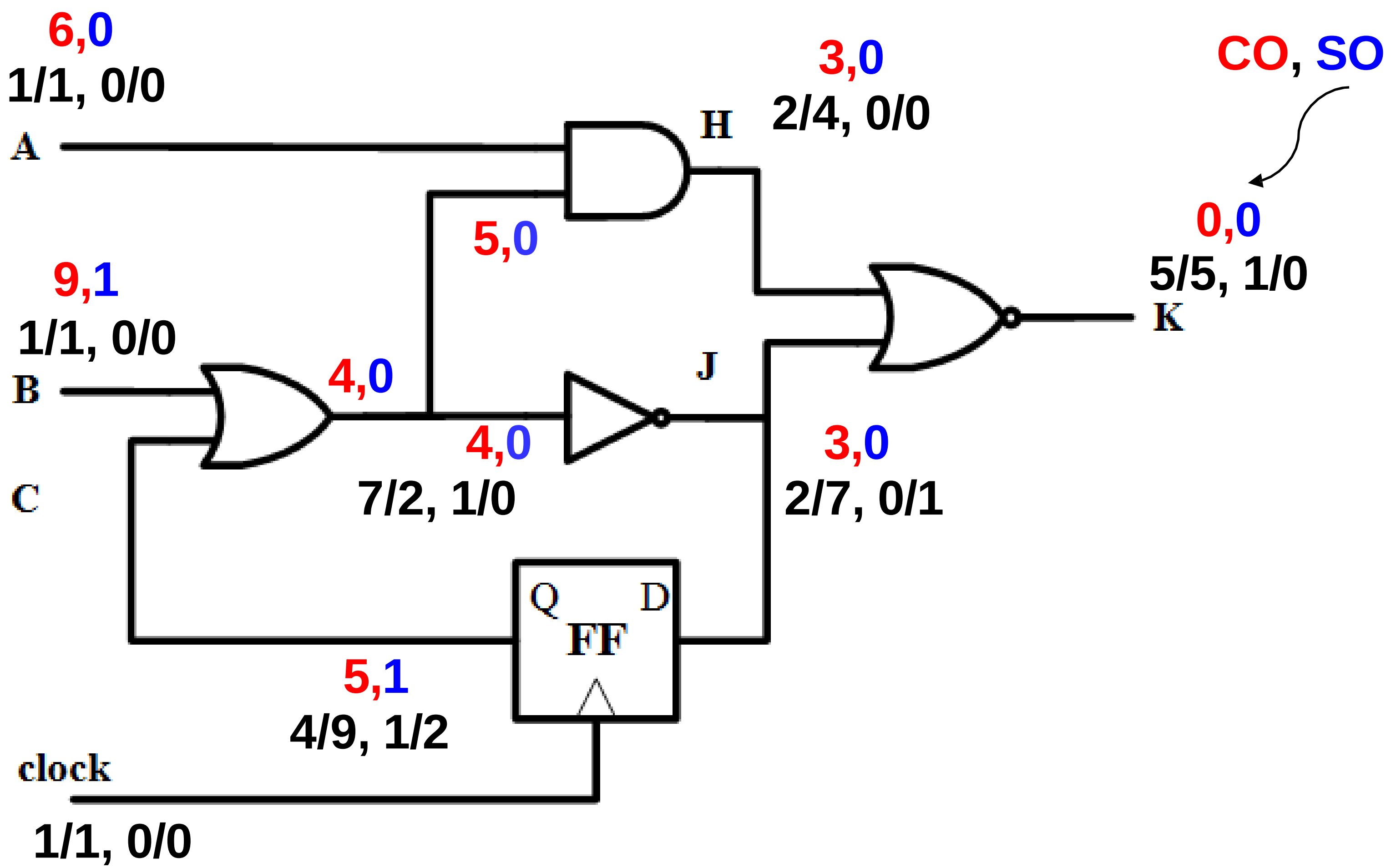


What Does SC^1=2 Mean?

- Control *C* to zero is easier. Assign *B*=1 and pulse one clock
- Control *C* to one is more difficult. Assign *B*=1 and *B*=0. Two clocks

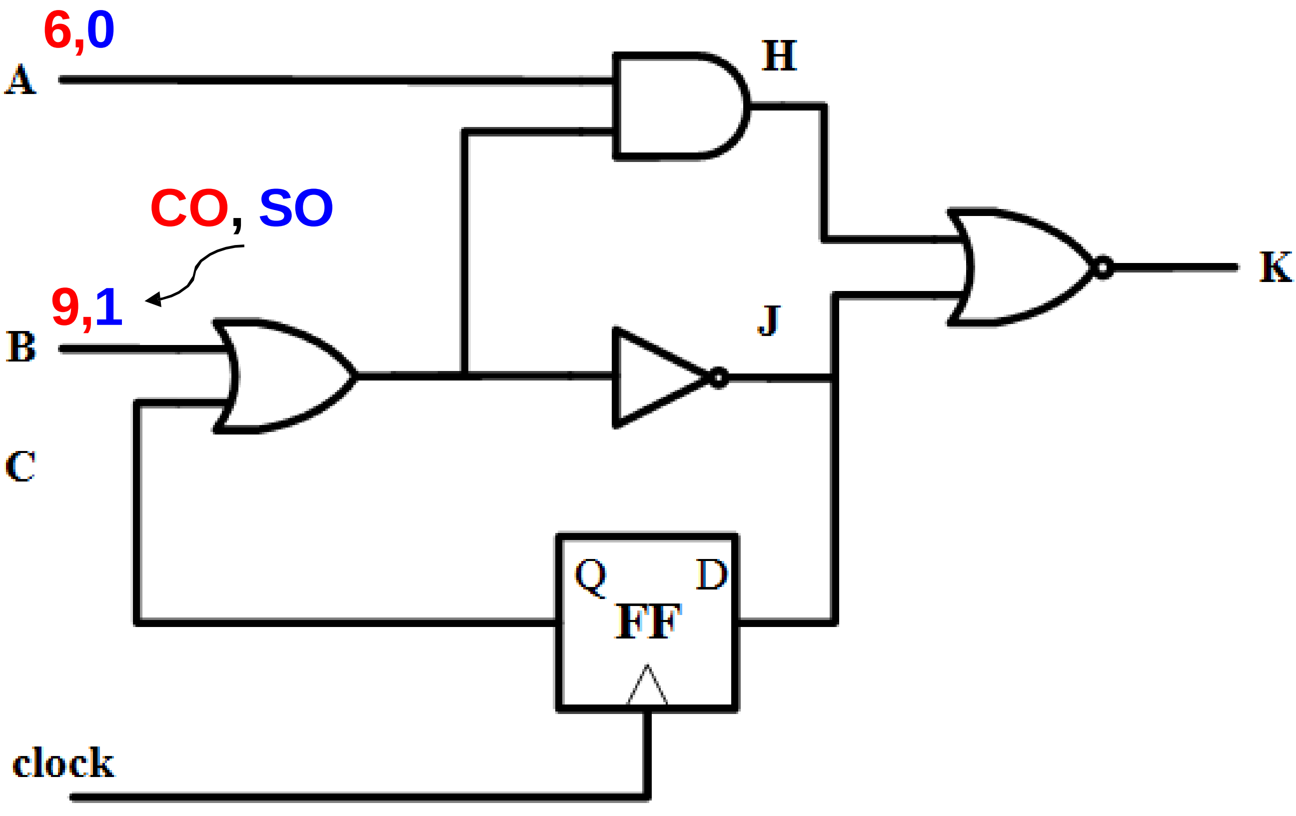


Obervability Computation



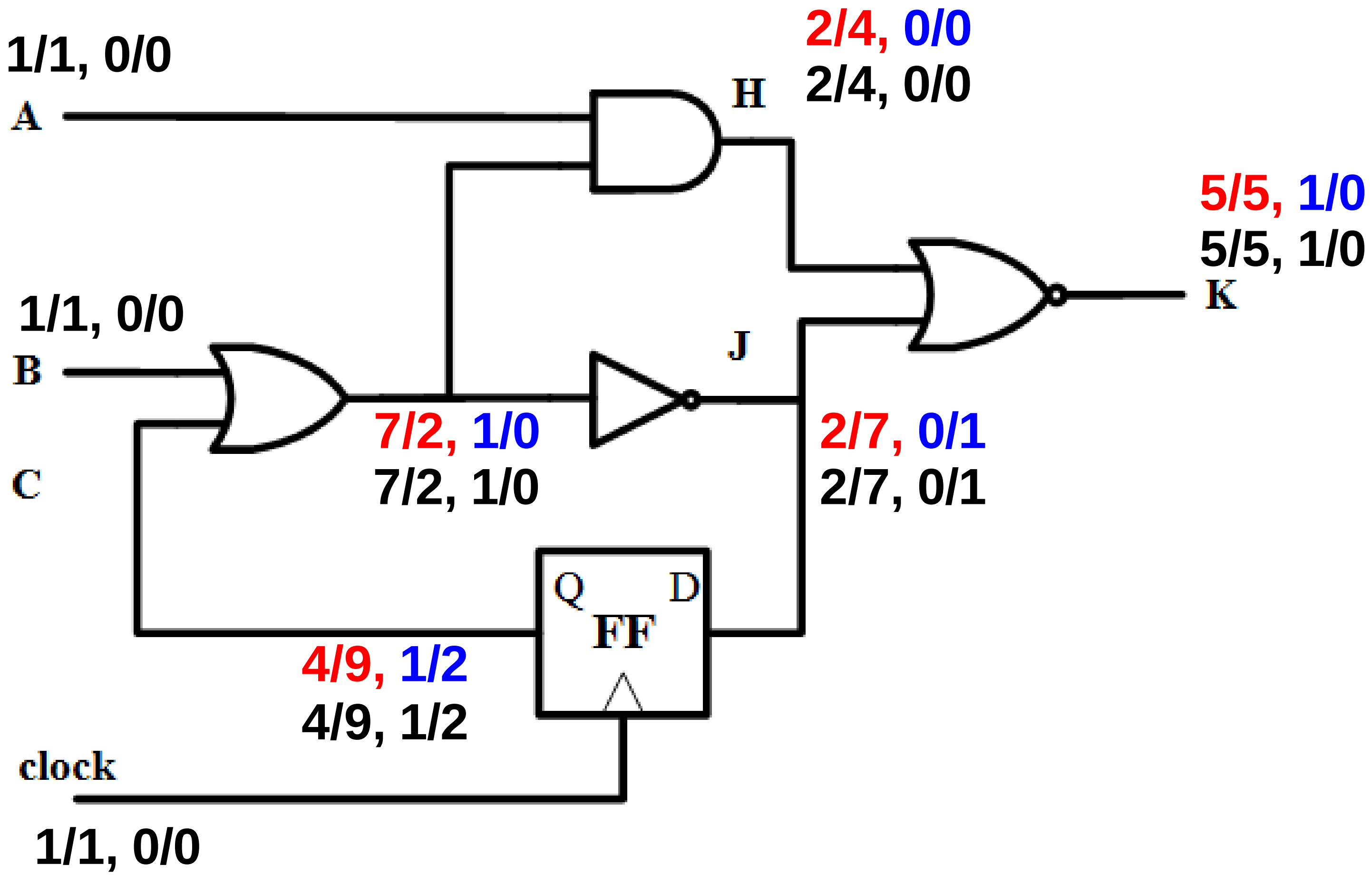
Quiz

Q1: Generate a sequence of test patterns to observe *A*?
Q2: Generate a sequence of test patterns to observe *B*?
(assume no scan. can only assign PI)



FFT

- When does the algorithm fail to converge?



Computing Sequential SCOAP

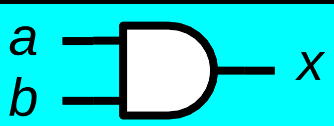
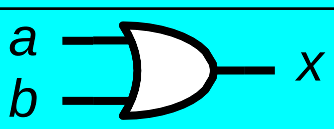
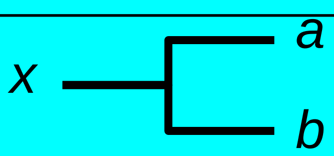
- Computation of $SC^0(N)$, $SC^1(N)$, and $SO(N)$ is similar to
  - $CC^0(N)$, $CC^1(N)$, and $CO(N)$.

- Differences are
  1. Increments sequential SCOAP by 1 only when signals propagate from FF inputs to Q, or backwards
  2. May require *iterations* for controllability to converge

COP

- *Signal probability* of *x* = probability of *x* being logic 1
  - Actual signal probability requires exhaustive simulation
  - Hard to obtain in practice
- COP = Controllability/Observability Program [Brglez 84]
  - Fast algorithm to estimate signal probability
  - C_x = estimated prob(x = *1)*
  - $1-C_x$ = estimated prob(x = 0)
  - O_x = estimated probability of *fault effect* in *x* being observed at PO
- C_x and O_x are numbers between 0 and 1
  - *Larger number* means *easier* to control or observe
- Assumptions
  - 1. Ignore fanout reconvergence for fast run time
  - 2. PI are independent random numbers: ½ zero and ½ one

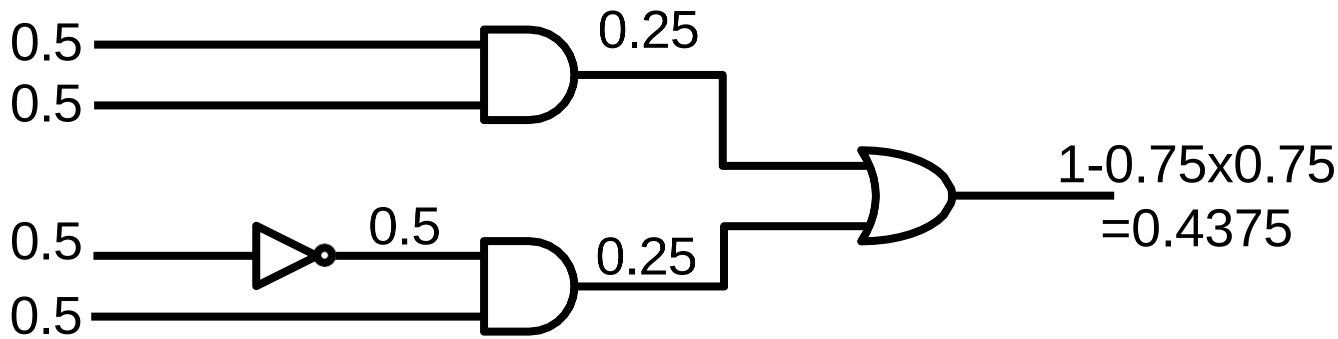
COP

	C_x	O_a
x = PI	*0.5*	
x = PO		*0.5*
	$C_x = C_a \times C_b$	$O_a = O_x \times C_b$
	$C_x = 1 - (1 - C_a) \times (1 - C_b)$	$O_a = O_x \times (1 - C_b)$
	$C_x = C_a = C_b$	$O_x = 1 - (1 - O_a) \times (1 - O_b)$

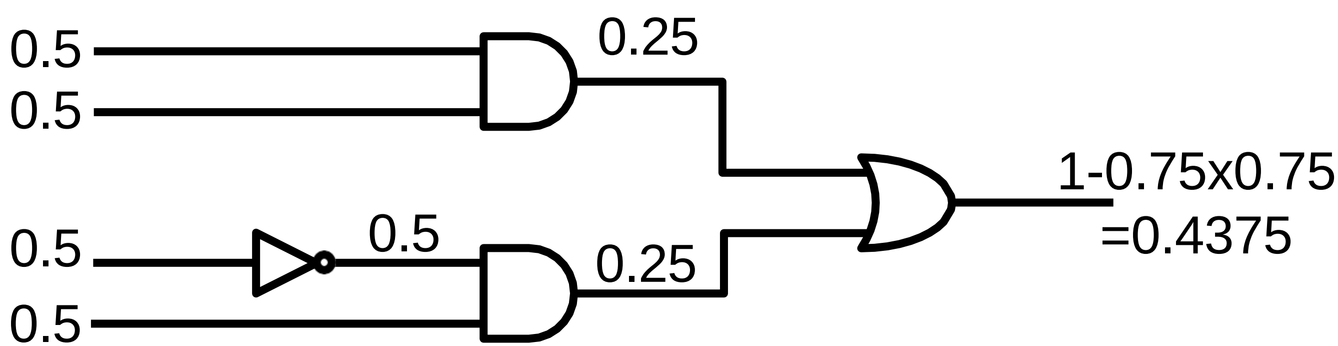
Example – Controllability

- **Calculate from PI to PO**
- **Fanout-free circuit, COP = actual signal probability**

COP C_x



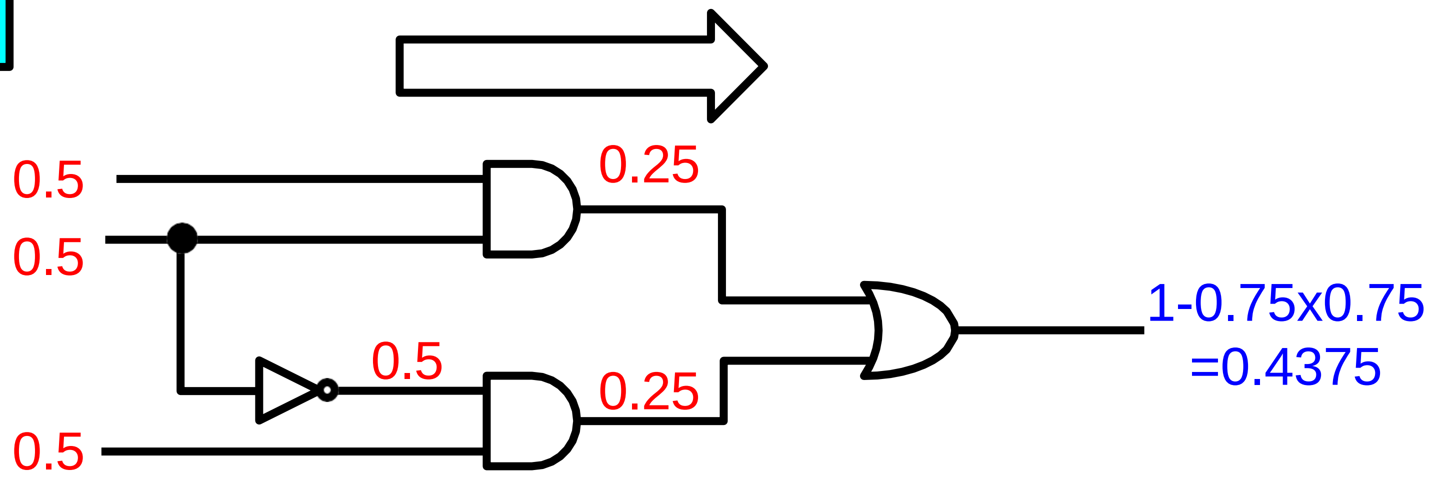
Actual signal probability



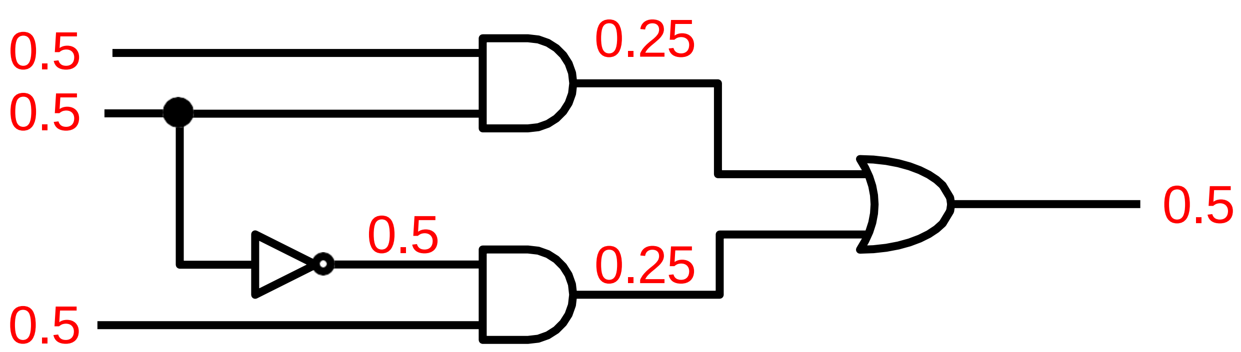
Example (2) – Controllability

- When **fanouts reconverge**, **COP $\neq$ actual signal probability**

COP C_x



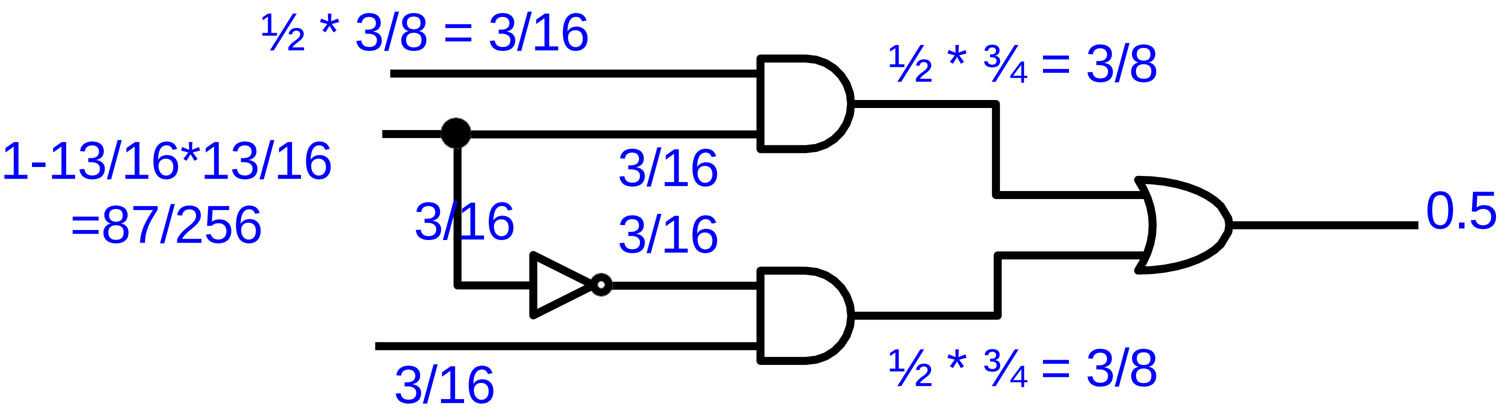
Actual signal probability



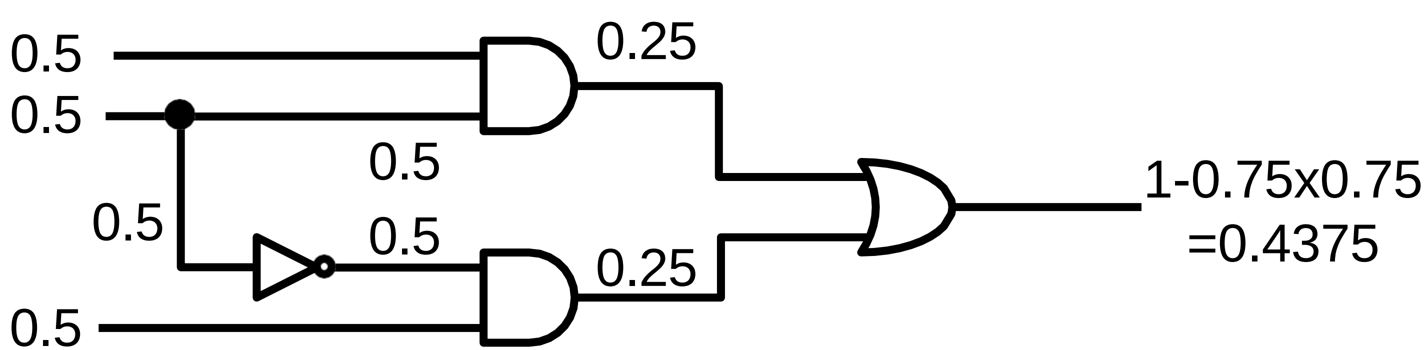
Example (2) – Observability

- **Calculate from PO to PI**

COP O_x



COP C_x

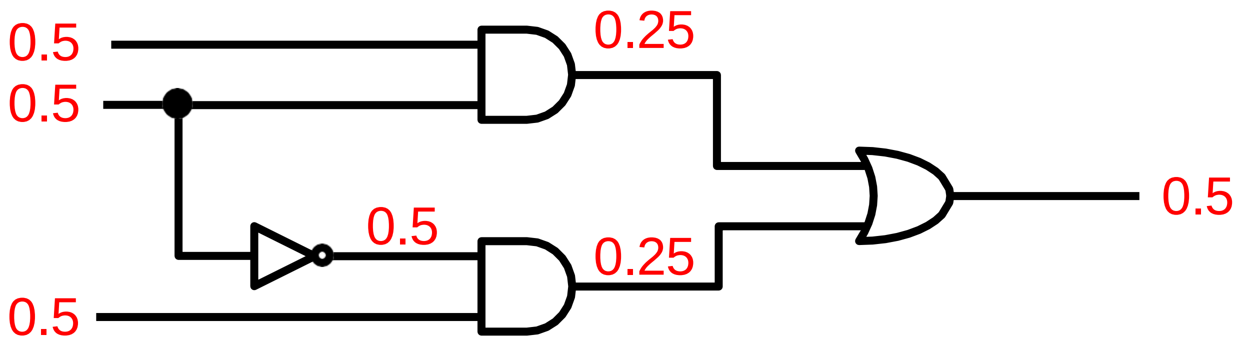


Quiz

Q: verify actual signal probability by exhaustive test patterns

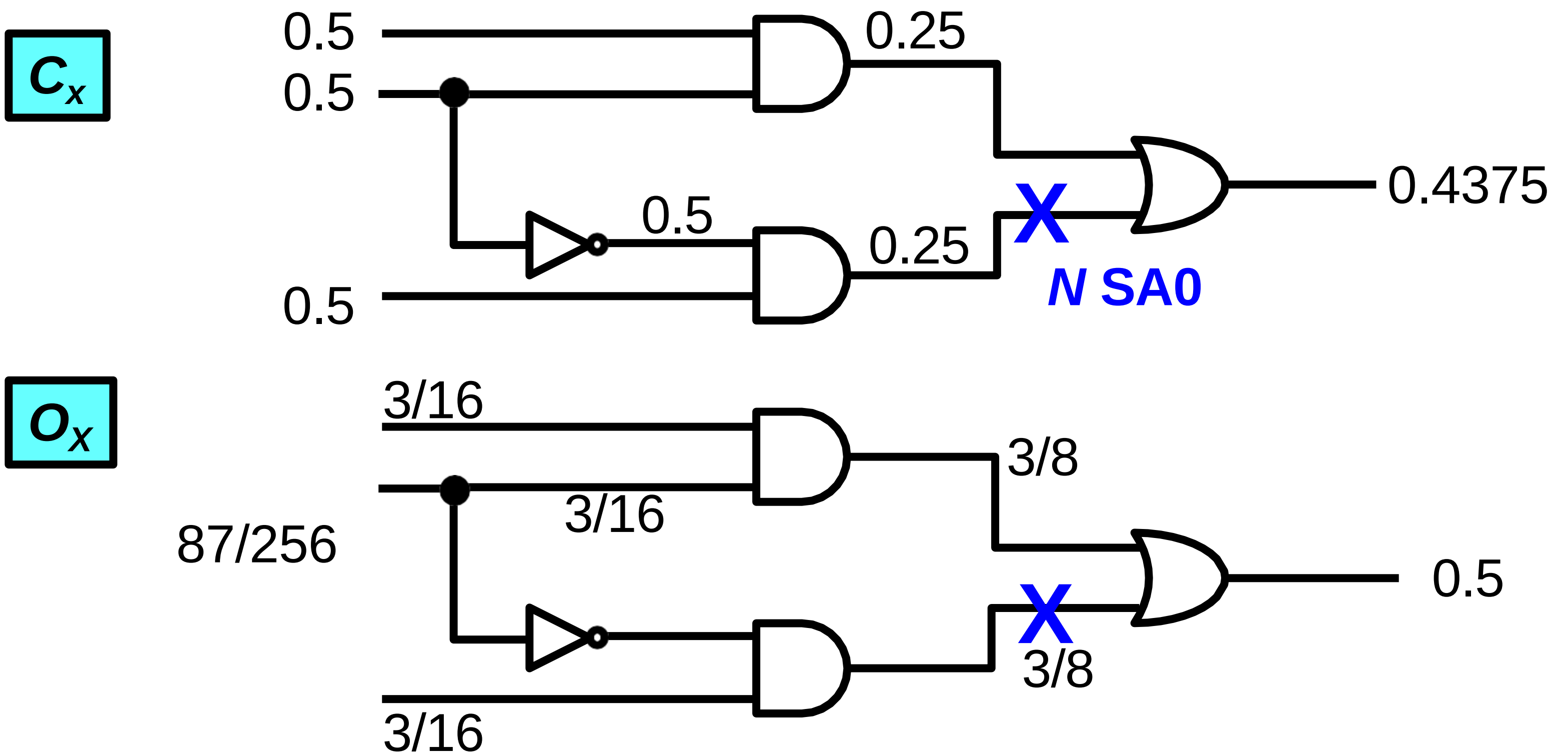
input	output
000	
001	
010	
011	
100	
101	
110	
111	

Actual signal probability



Detection Probability, DP

- ***DP_f*** **= Probability of detecting a fault *f***
  - **$DP_{N\ SA0} = C_N \times O_N$**
  - **$DP_{N\ SA1} = (1-C_N) \times O_N$**
- **Larger *DP_f* means *easier* to detect fault *f***
- **Example: $DP_{N\ SA0}$ = 1/4 x 3/8 =3/32**

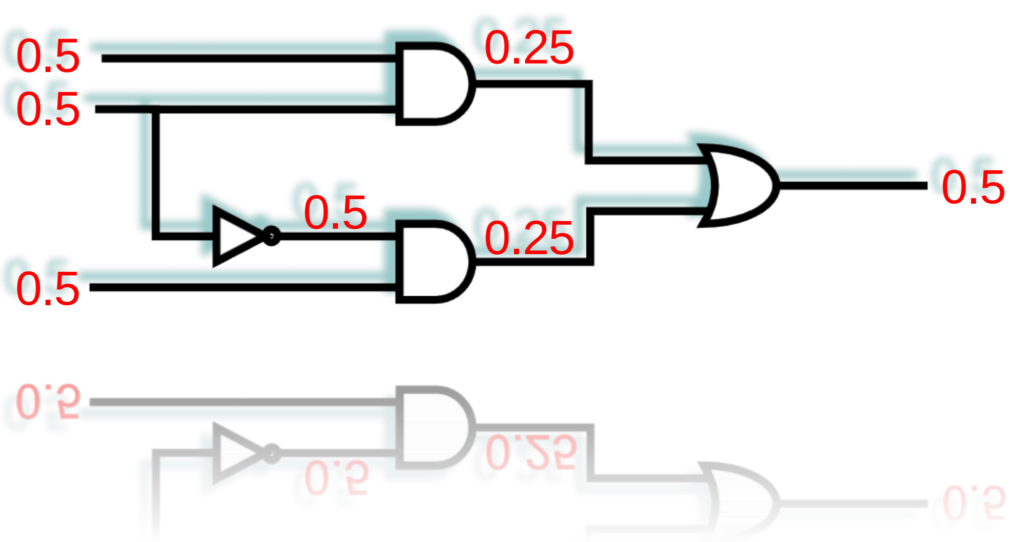


Random Pattern Resistant Faults

- **RPRF = Faults that are difficult to be tested by random patterns**
  - ***Low detectability***
  - **aka. *Hard-to-detect faults, difficult faults***
- **Example:**
  - **stuck-at-0 fault at an *n*-input AND gate output**
  - **Need test pattern (1,1,1,…1)**
  - **Assume equal signal probability of 0.5 at each input**
    - $C_x = 0.5^n$
- **Test generation for RPRF is difficult**
  - **Solutions:**
    - **1. Insert test points (See DFT lecture)**
    - **2. Weighted random patterns (see BIST lecture)**

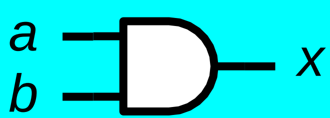
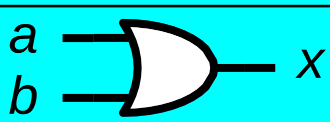
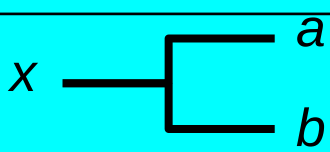
Summary

- **<u>COP</u>**
  - **C_x = estimated prob(x = *1*)**
  - **$1\text{-}C_x$ = estimate prob(x = 0)**
  - **O_x = estimated probability of *fault effect* in *x* being observed**
  - **COP ≠ actual signal probability because fanout reconvergence**



FFT

- **Q: Why observability at PO is 0.5, not 1?**

	C_x	O_a
***x* = PI**	***0.5***	
***x* = PO**		***0.5***
	$C_x = C_a \times C_b$	$O_a = O_x \times C_b$
	$C_x = 1 - (1 - C_a) \times (1 - C_b)$	$O_a = O_x \times (1 - C_b)$
	$C_x = C_a = C_b$	$O_x = 1 - (1 - O_a) \times (1 - O_b)$