

# More Details on Faults - Organization

- Fault Modeling
  - ⇒ Equivalent Faults & Collapsing
  - ⇒ Gate Level Faults
  - ⇒ Transistor Level Faults
  - ⇒ Bridging Faults
- Fault Detection
  - ⇒ Gate Level Faults
  - ⇒ Transistor Level Faults
  - ⇒ Bridging Faults
  - ⇒ Fault Simulations
    - ⊙ Undetectable & Potentially Detected
    - ⊙ Fault Coverage

# Fault Modeling

## *Recall:*

- A good fault model has 2 requirements:
  1. accurately reflects the behavior of a physical defect
  2. is computationally efficient with respect to simulation
- Single fault models are used for requirement # 2
- The most commonly used current fault models include:
  - ⇒ Gate level stuck-at faults
    - ⊙ stuck-at-0 (sa0) & stuck-at-1 (sa1)
  - ⇒ Transistor level stuck faults
    - ⊙ stuck-on (stuck-closed) & stuck-off (stuck-open)
  - ⇒ Bridging faults (shorts between wires)
    - ⊙ wired-AND & wired-OR
    - ⊙ dominant (one driving source dominates the other)
      - ↳ Note: opens in wires typically covered by stuck-faults

# Gate Level Stuck-at Fault Model

- Gate inputs or outputs can be:

⇒ Stuck-at-0 (sa0)

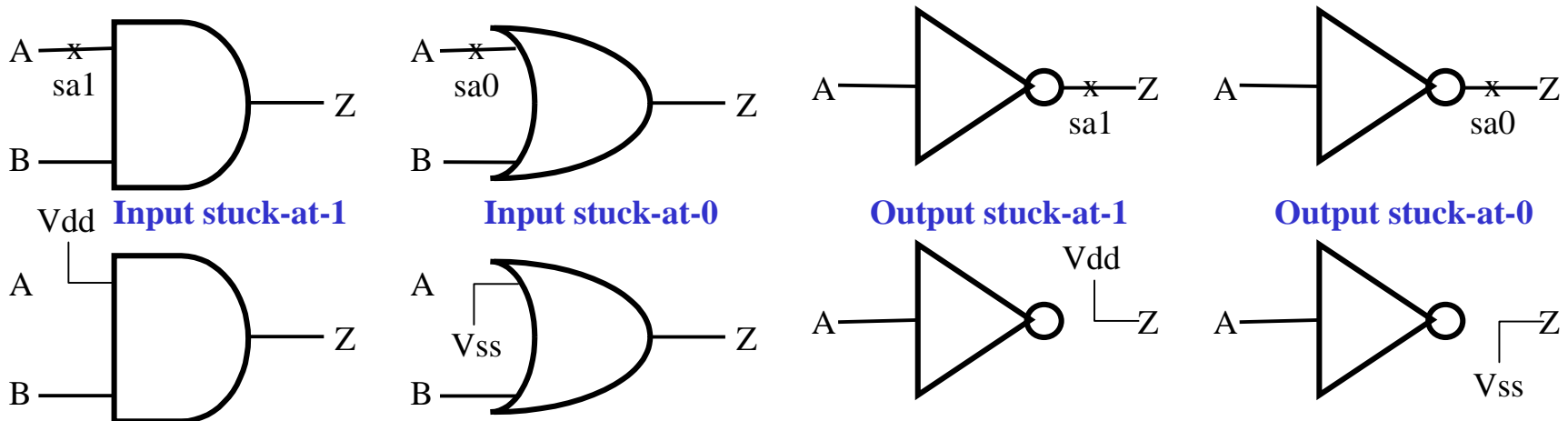
⊙ as if input or output were disconnected and tied low to  $V_{ss}$

⇒ Stuck-at-1 (sa1)

⊙ as if input or output were disconnected and tied high to  $V_{dd}$

⇒ fault site denoted by 'X' with sa0/sa1

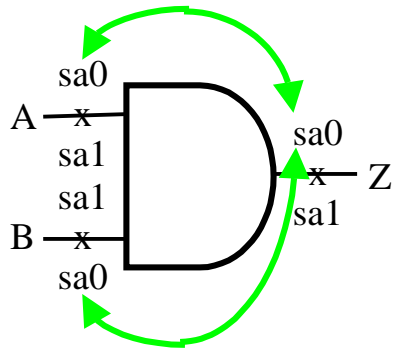
⊙ Note: there is no feedback of fault value from fault site!



# Gate Level Equivalent Faults and Fault Collapsing

- Equivalent faults are indistinguishable & can be collapsed
  - ⇒ 1 fault in set of equivalent faults represents all in set
  - ⇒ Fewer faults to simulation ⇒ faster fault simulations
- Gate level collapsing
  - ⇒ **AND** (**NAND**) gates
    - ⊙ any input  $sa0 =$  output **sa0** (**sa1**)
      - ↳ # Collapsed Faults =  $I + 2$  (where  $I = \#$  inputs)
  - ⇒ **OR** (**NOR**) gates
    - ⊙ any input  $sa1 =$  output **sa1** (**sa0**)
      - ↳ # Collapsed Faults =  $I + 2$  (where  $I = \#$  inputs)
  - ⇒ **INVERTER**
    - ⊙ input **sa0** (**sa1**) = output **sa1** (**sa0**)
      - ↳ # Collapsed Faults = 2

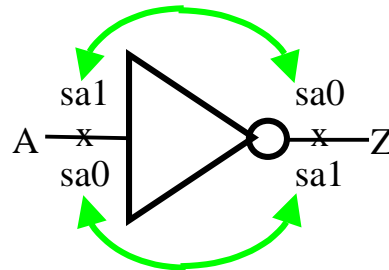
# Gate Level Equivalent Faults and Fault Collapsing



Collapsed fault set  
 $Z\ sa0 = A\ sa0 = B\ sa0$   
 $Z\ sa1$   
 $A\ sa1$   
 $B\ sa1$

AND

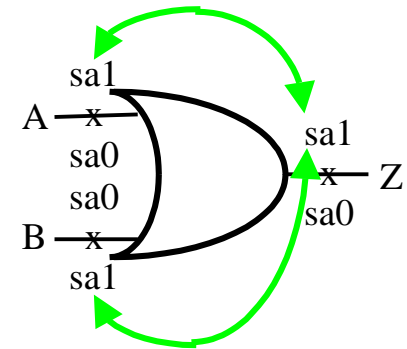
AB	Z	Asa0	Asa1	Bsa0	Bsa1	Zsa0	Zsa1
00	0	0	0	0	0	0	1
01	0	0	1	0	0	0	1
10	0	0	0	0	1	0	1
11	1	0	1	0	1	0	1



Collapsed fault set  
 $Z\ sa1 = A\ sa0$   
 $Z\ sa0 = A\ sa1$

INVERTER

A	Z	Asa0	Asa1	Zsa0	Zsa1
0	1	1	0	0	1
1	0	0	1	1	0



Collapsed fault set  
 $Z\ sa1 = A\ sa1 = B\ sa1$   
 $Z\ sa0$   
 $A\ sa0$   
 $B\ sa0$

OR

AB	Z	Asa0	Asa1	Bsa0	Bsa1	Zsa0	Zsa1
00	0	0	1	0	1	0	1
01	1	1	1	0	1	0	1
10	1	0	1	1	1	0	1
11	1	1	1	1	1	0	1

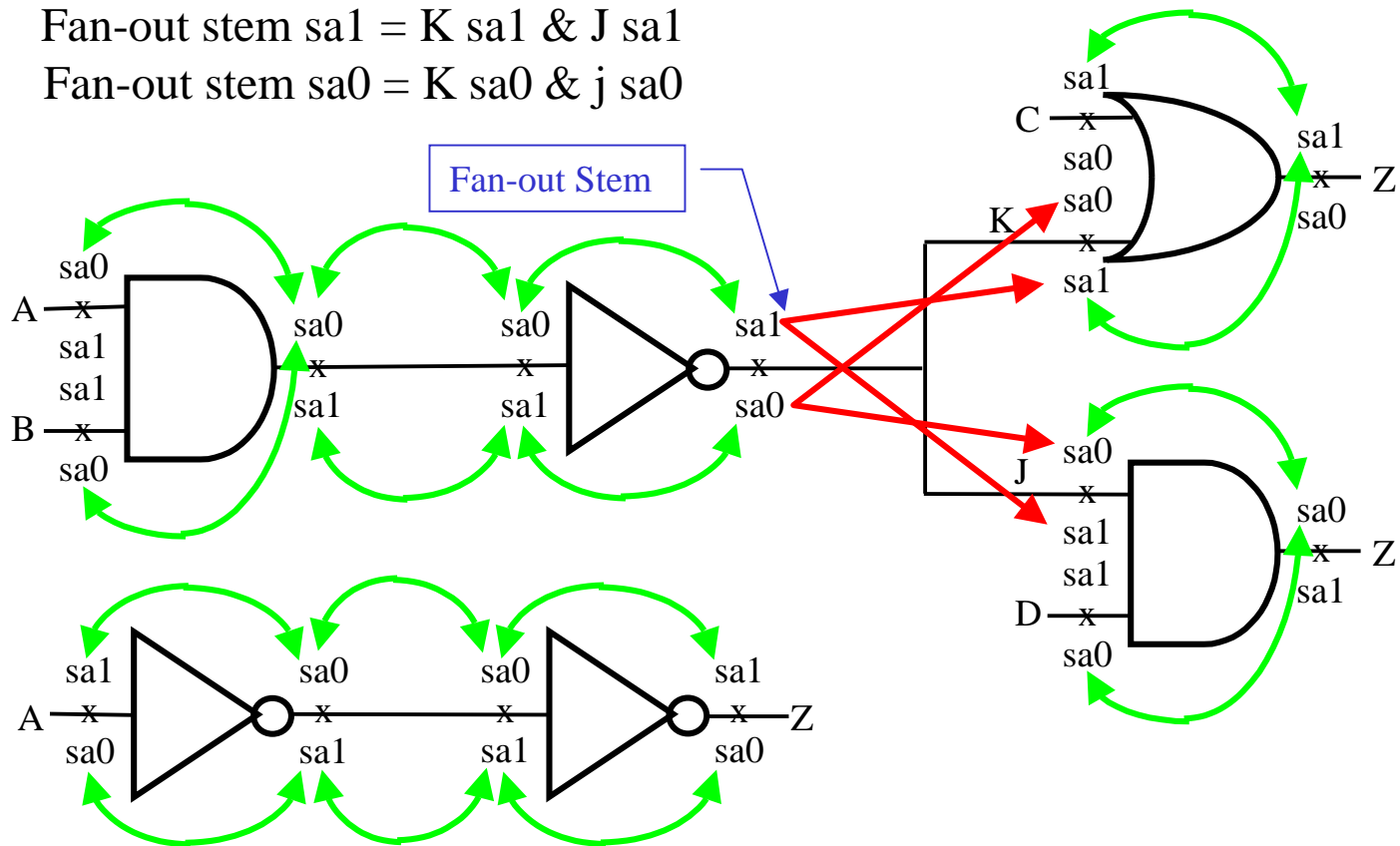
# Structural Equivalent Faults and Fault Collapsing

Cannot collapse faults at fan-out stem since it would violate single stuck-at fault model

Fan-out stem  $sa1 = K sa1 \& J sa1$

Fan-out stem  $sa0 = K sa0 \& j sa0$

# collapsed faults = 12

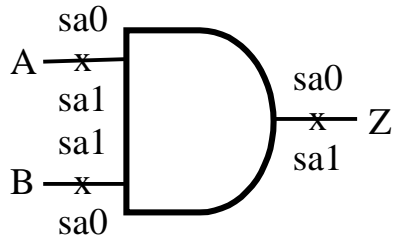


Only 2 collapsed faults for inverter chain ( $Z sa0 \& Z sa1$ )

# Collapsed vs. Uncollapsed Gate Level Fault Sets

- For a given circuit (assuming elementary logic gates)
  - ⇒ # uncollapsed faults =  $2(G + G_I)$ 
    - ⊙  $G$  = total # gates
    - ⊙  $G_I$  = total # gate inputs
  - ⇒ # collapsed faults =  $2(O_P + F_S) + G_I - N_I$ 
    - ⊙  $O_P$  = # primary outputs
    - ⊙  $F_S$  = # fan-out stems
    - ⊙  $N_I$  = # inverters
  - ⇒ typically #collapsed flts  $\approx 1/2$  # uncollapsed flts
- Should faults be collapsed?
  - ⇒ YES: for TPG and fault simulation (more efficient)
  - ⇒ NO: for computing fault coverage (more accurate)
    - ⊙ but longer fault simulation times

# Gate Level Fault Detection



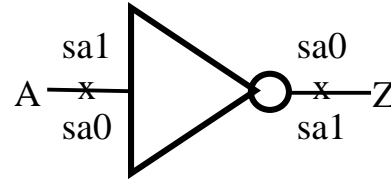
Collapsed fault set  
 $Z_{sa0} = A_{sa0} = B_{sa0}$   
 $Z_{sa1}$   
 $A_{sa1}$   
 $B_{sa1}$

AND

AB	Z	A <sub>sa0</sub>	A <sub>sa1</sub>	B <sub>sa0</sub>	B <sub>sa1</sub>	Z <sub>sa0</sub>	Z <sub>sa1</sub>
00	0	0	0	0	0	0	1
01	0	0	1	0	0	0	1
10	0	0	0	0	1	0	1
11	1	0	1	0	1	0	1

Minimum Set of Test Vectors

01  
 10  
 11



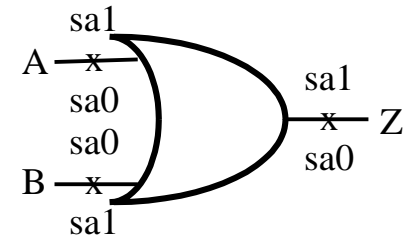
Collapsed fault set  
 $Z_{sa1} = A_{sa0}$   
 $Z_{sa0} = A_{sa1}$

INVERTER

A	Z	A <sub>sa0</sub>	A <sub>sa1</sub>	Z <sub>sa0</sub>	Z <sub>sa1</sub>
0	1	1	0	0	1
1	0	0	1	1	0

Minimum Set of Test Vectors

0  
 1



Collapsed fault set  
 $Z_{sa1} = A_{sa1} = B_{sa1}$   
 $Z_{sa0}$   
 $A_{sa0}$   
 $B_{sa0}$

OR

AB	Z	A <sub>sa0</sub>	A <sub>sa1</sub>	B <sub>sa0</sub>	B <sub>sa1</sub>	Z <sub>sa0</sub>	Z <sub>sa1</sub>
00	0	0	1	0	1	0	1
01	1	1	1	0	1	0	1
10	1	0	1	1	1	0	1
11	1	1	1	1	1	0	1

Minimum Set of Test Vectors

00  
 01  
 10

# Minimum Set of Test Vectors for Gate Level Faults

- Inverter requires both input logic values (0 and 1)
  - ⇒ # vectors = 2
- $N$ -input AND or NAND gate
  - ⇒ # vectors =  $N + 1$ 
    - ⊙ all 1s
    - ⊙ walk 0 through a field of 1s
- $N$ -input OR or NOR gate
  - ⇒ # vectors =  $N + 1$ 
    - ⊙ all 0s
    - ⊙ walk 1 through a field of 0s
- XOR  $\neq$  elementary logic gate (*made from multiple gates*)
  - ⇒ # faults depends on construction of gate
    - ⊙ 3 vectors required for pin faults
    - ⊙ all 4 vectors required for internal faults

# Transistor Level Fault Model

- Transistor can be:

⇒ Stuck-on (a.k.a. stuck-short)

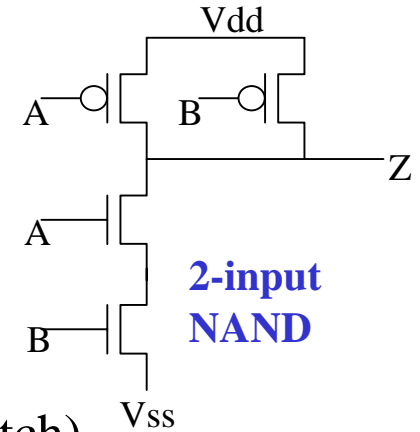
⊙ can result in excessive  $I_{DDQ}$

⇒ Stuck-off (a.k.a. stuck-open)

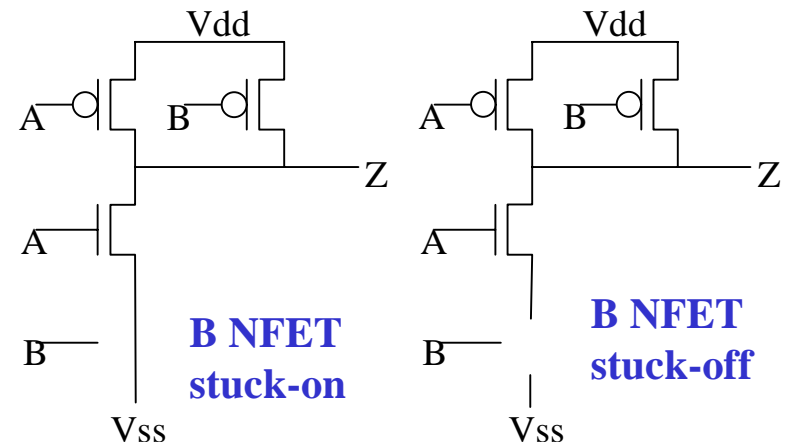
⊙ can result in “memory” node (logic gate ⇒ latch)

- Gate level fault accurate for NMOS but not for CMOS

⇒ gate input stuck-at = 2 transistors stuck-at in CMOS



		A PFET		A NFET		B PFET		B NFET	
AB	Z	s-on	s-off	s-on	s-off	s-on	s-off	s-on	s-off
00	1	1	1	1	1	1	1	1	1
01	1	1	mem	$I_{DDQ}$	1	1	1	1	1
10	1	1	1	1	1	1	mem	$I_{DDQ}$	1
11	0	$I_{DDQ}$	0	0	mem	$I_{DDQ}$	0	0	mem



# Transistor Level Fault Equivalence & Collapsing

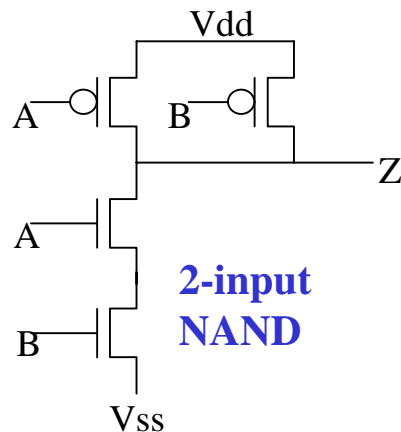
- Stuck-off faults in series transistors are equivalent
- Stuck-on faults in parallel transistors are equivalent
- # collapsed transistor faults =  $2T - N_{ser} + G_{ser} - N_{par} + G_{par}$

↳  $N_{ser}$  = total # series transistors

↳  $G_{ser}$  = total # groups of series transistors

↳  $N_{par}$  = total # parallel transistors

↳  $G_{par}$  = total # groups of parallel transistors



# collapsed faults = 6

A PFET s-on = B PFET s-on

A NFET s-off = B NFET s-off

A PFET s-off

B PFET s-off

A NFET s-on

B NFET s-on

# Transistor Level Fault Detection

- Transistor fault detection more difficult than gate level

⇒ Stuck-on faults

- ⊙ voltage divider may not produce incorrect logic values

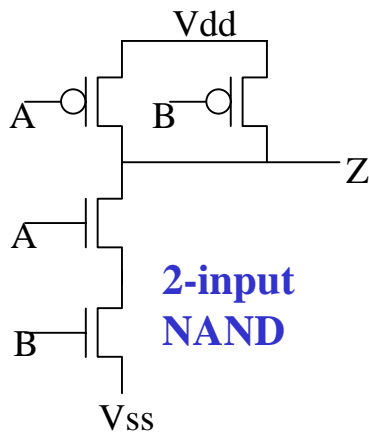
- ⊙ monitoring  $I_{DDQ}$  is best approach

- ↳ small currents may be lost in leakage of  $>2M$  transistors

⇒ Stuck-off can

- ⊙ will produce wrong logic values

- ↳ need ordered set of 2 vectors to detect



		A PFET		A NFET		B PFET		B NFET	
AB	Z	s-on	s-off	s-on	s-off	s-on	s-off	s-on	s-off
00	1	1	1	1	1	1	1	1	1
01	1	1	mem $I_{DDQ}$	1	1	1	1	1	1
10	1	1	1	1	1	mem $I_{DDQ}$	1	1	1
11	0	$I_{DDQ}$	0	0	mem $I_{DDQ}$	$I_{DDQ}$	0	0	mem

**Vectors to detect:**

A PFET stuck-off

11 - to get Z=0

01 - to detect Z=0

B PFET stuck-off

11 - to get Z=0

10 - to detect Z=0

A/B NFETs s-off

0x or x0 - to get Z=1

11 to detect Z=1

Min. #vectors = 4

detects s-on w/  $I_{DDQ}$

# Bridging Fault Models

- Two current models for wires shorted together:

⇒ **Wired-AND/Wired-OR** fault model

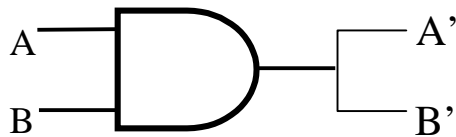
- Shorted wires perform logical AND or OR

⇒ **Dominant** fault model

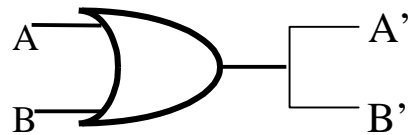
- Stronger driving gate dominates the short

- For  $N$  nets, # pair-wise bridging faults =  $N^2 - N$  (either model)

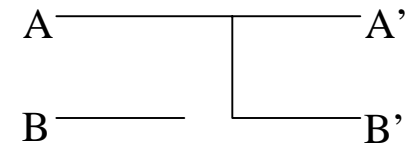
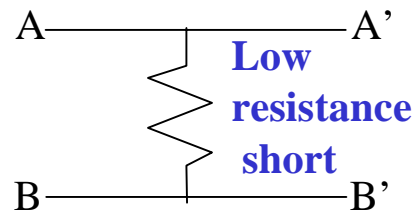
- No fault equivalence ⇒ no fault collapsing



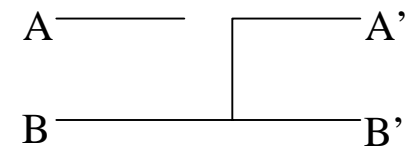
**Wired-AND fault model**



**Wired-OR fault model**



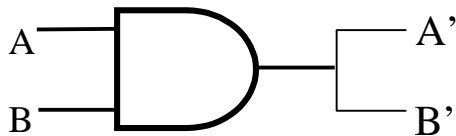
**A dominates B model**



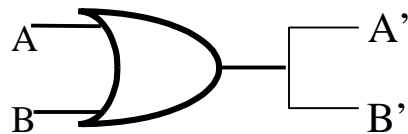
**B dominates A model**

# Bridging Fault Detection

- Wired-AND/Wired-OR faults
  - ⇒ 1 vector (01 or 10) with 2 outputs (A' and B'), *or*
  - ⇒ 1 output (A' or B') with 2 vectors (01 and 10)
- Dominant faults
  - ⇒ 1 vector (01 or 10) with 2 outputs (A' and B')
  - ↳ harder to detect than Wired-AND/OR (*less observable*)
  - ⊙ detecting all dominant BFs ⇒ detecting all Wired-AND/OR

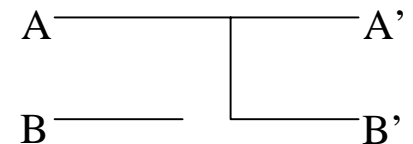


Wired-AND fault model

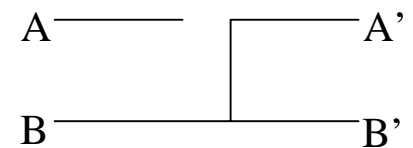


Wired-OR fault model

AB	A'B'	WAND	WOR	A <sub>dom</sub> B	B <sub>dom</sub> A
00	00	00	00	00	00
01	01	00	11	00	11
10	10	00	11	11	00
11	11	11	11	11	11



A dominates B model



B dominates A model

# Fault Detection

## **Recall:**

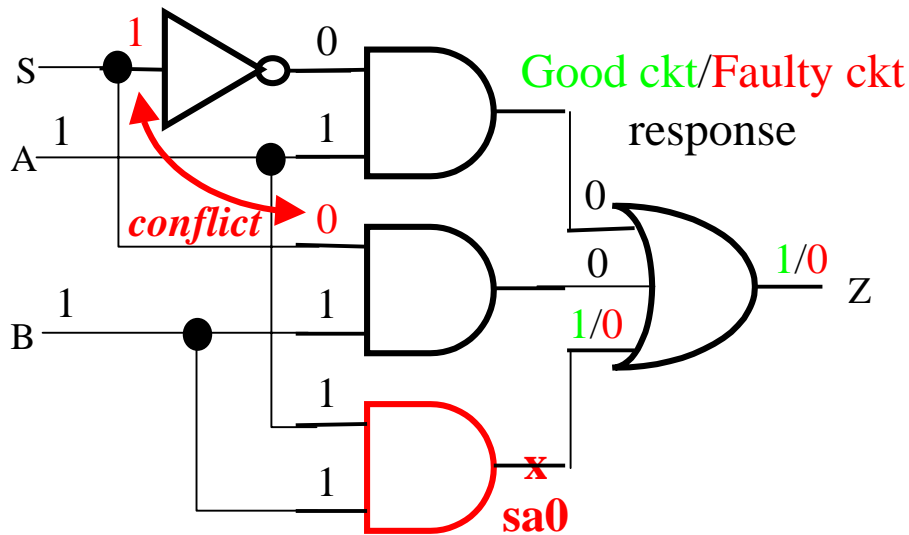
- Fault detection requires:
  - ⇒ observation of an error (from fault) at a primary output
    - ⊙ **observability** of the fault site
      - ↳ the ease at which we can observe the fault behavior
  - ⇒ input stimuli that creates an error as a result of fault
    - ⊙ **controllability** of the fault site
      - ↳ the ease at which we can control the fault behavior
    - ⊙ **controllability** of path from fault site to primary output
      - ↳ typically considered part of observability
- Testability  $\propto$  **controllability** & **observability**
- Any given fault may be:
  - ⇒ Detectable
  - ⇒ Undetectable
  - ⇒ Potentially detectable

# Gate Level Fault Detection - Path Sensitization

1. At fault site, assign logic value opposite that of stuck-at fault
  2. From fault site, choose a path to a PO assigning non-controlling values to all other inputs to gates in that path
    - 1 = non-controlling value for AND/NAND gates
    - 0 = non-controlling value for OR/NOR gates
  3. For all assigned values, back-trace to PIs selecting input values that will produce the assigned values
  4. If there is a conflict, repeat Steps 2 & 3 choosing new paths and/or values in Step 3
- If no path can be found without conflict, the fault *may be* undetectable, otherwise values at PIs form test vector

# Undetectable Faults

- No test vector can detect the fault
  - ⇒ usually difficult to prove a fault is undetectable
- Undetectable faults due to:
  - ⇒ Re-convergent fan-out, *and*
  - ⇒ Redundant logic



Hazard-free multiplexer has undetectable faults due to re-convergent fanout and *redundancy*

		AB			
		00	01	11	10
S	0	0	0	1	1
	1	0	1	1	0

## Undetectable Faults (cont.)

- Minimize circuit to remove redundancy & these faults
  - ⇒ undetectable fault may not effect circuit operation
    - ⊙ wasted area
  - ⇒ undetectable fault slows down fault simulation
    - ⊙ all test vectors are simulated (no trip on mismatch)
- Sometimes undetectable faults are unavoidable
  - ⇒ hazard-free circuits
    - ⊙ glitch-free clock multiplexing
  - ⇒ initialization circuitry
    - ⊙ power-up presets
    - ⊙ global resets
  - ⇒ use these ckts sparingly to minimize undetectable faults

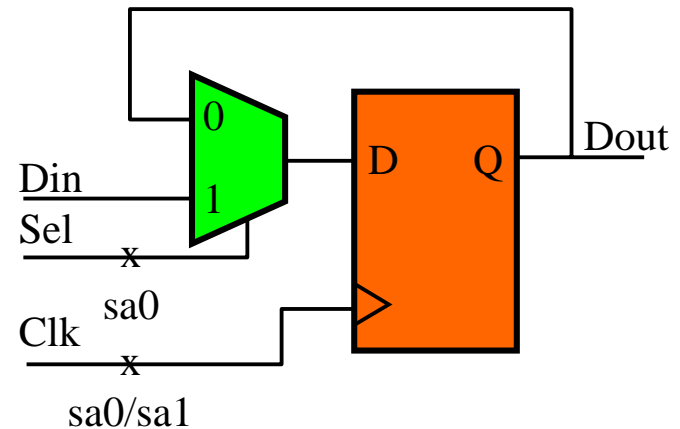
# Potentially Detected Faults

- Due to undefined logic values ( $U, 2, X$ )
  - ⇒ an artifact of logic simulation
    - ⊙ can be a 0 or a 1 but simulator doesn't know
    - ⊙ used for un-initialized logic
  - ⇒ faults preventing initialization produce  $U, 2, X$
- Potential detect fault if good ckt = 1/0 & faulty ckt =  $U, 2, X$ 
  - ⇒ potential detect faults may be detected by other vectors
  - ⇒ probability of detect of a fault  $\propto$  # potential detects
    - ⊙ for high data activity, otherwise probability = 0.5
  - ⇒ in real circuit, fault may/may not be detected
    - ⊙ depends on power-up value
  - ⇒ in simulation, PDFs also show up as undetected faults

# Potentially Detected Faults (cont.)

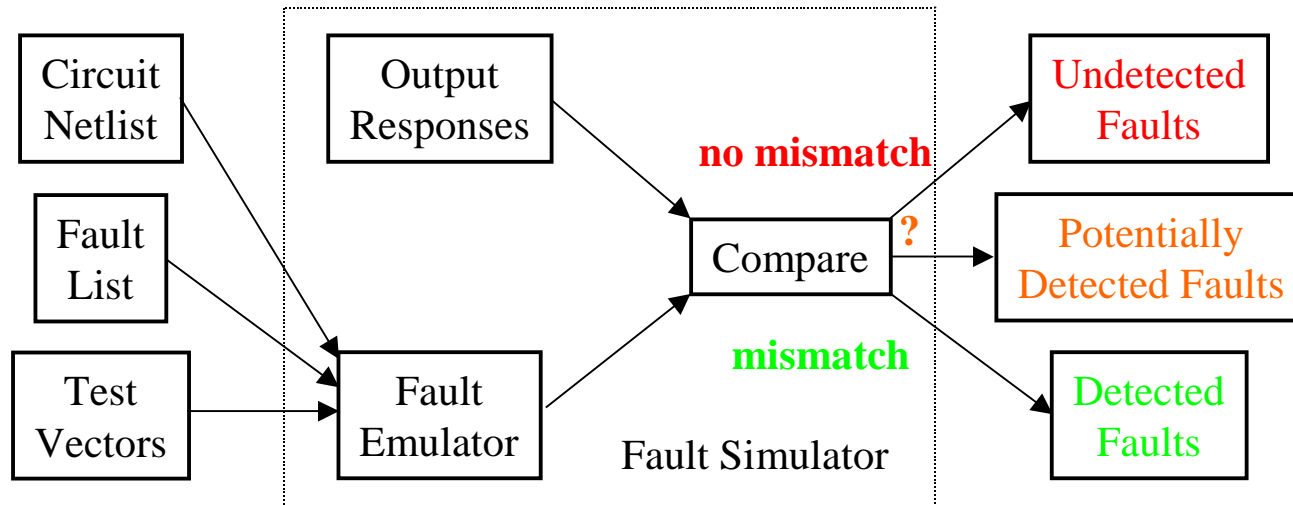
## Examples of potentially detected fault

- Select input to MUX stuck-at-0
  - ⇒ flip-flop cannot be initialized
- Clock stuck-at-0 and stuck-at-1
  - ⇒ flip-flop cannot be initialized
- 3 potentially detected faults
  - ⇒ high detection probability for
    - ⊙ high data activity on Sel & Din
    - ⊙ high clock frequency
  - ⇒ otherwise these may not be detected



# Fault Simulation

- Fault simulator emulates faults and compares resultant response to known good circuit output responses
- Fault simulation long for large fault lists - speed-up:
  - ⇒ simulation of a given faults ends on detection
  - ⇒ parallel flt simulation emulates 1 flt/bit (computer word)
  - ⇒ statistical fault sampling (>1000 samples = good estimate)



# Fault Coverage/Grading

- Given a set of test vectors, each fault in fault set can be:

⇒  $D = \text{detected faults}$

- ⊙ Targeted faults and faults “accidentally” detected

⇒  $X = \text{undetectable faults}$

- ⊙ There are **NO** vectors that can detect these faults

⇒  $U = \text{undetected faults}$

- ⊙ Could not find vector to detect fault (*but there could be one*)

⇒  $P = \text{potentially detected faults (PDF)}$

- ⊙ Also included in  $U$

⇒  $T = \text{total faults} = D + X + U$

- Fault coverage =  $(D + P/2) / T$

⇒ Detectable FC =  $(D + P/2) / (T - X)$

- ⊙ Note: assumes PDF detection probability = 0.5

## Need to Add

- Dominant-AND/OR BF model
- N-detectability