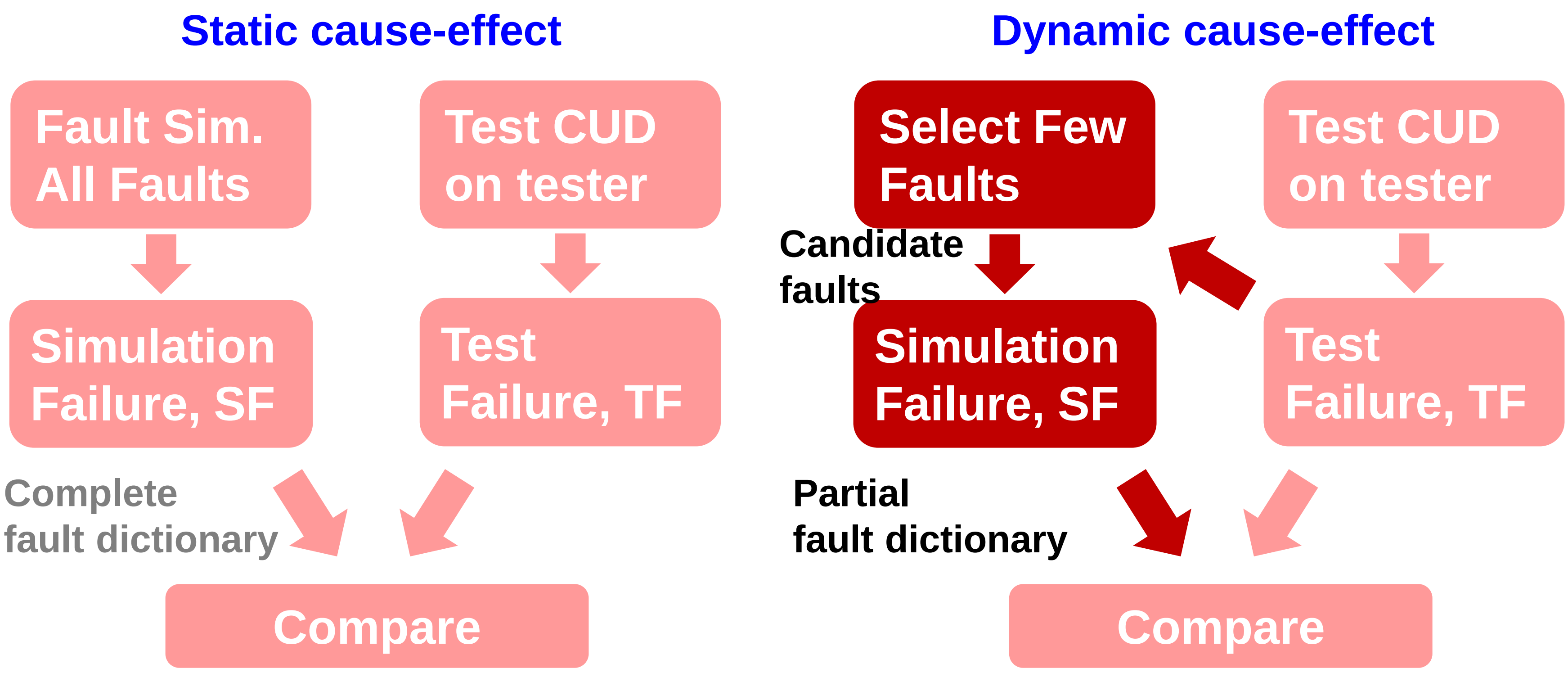
Diagnosis

- Introduction
- <u>Logic Diagnosis</u>
  - <u>SSF diagnosis</u>
    - Static Cause-effect diagnosis
    - <u>Dynamic Cause-effect diagnosis</u>
    - Effect-cause diagnosis
  - Unmodeled / multiple fault diagnosis
- Scan Chain Diagnosis
- Failure Analysis
- Conclusions

Dynamic Cause-Effect Diagnosis

- "Dynamic" because fault dictionary changes with TF
- Procedure:
  - 1. Test CUD, then select a few candidate faults
  - 2. Fault simulate to generate *partial fault dictionary*



Which Faults to be Simulated?

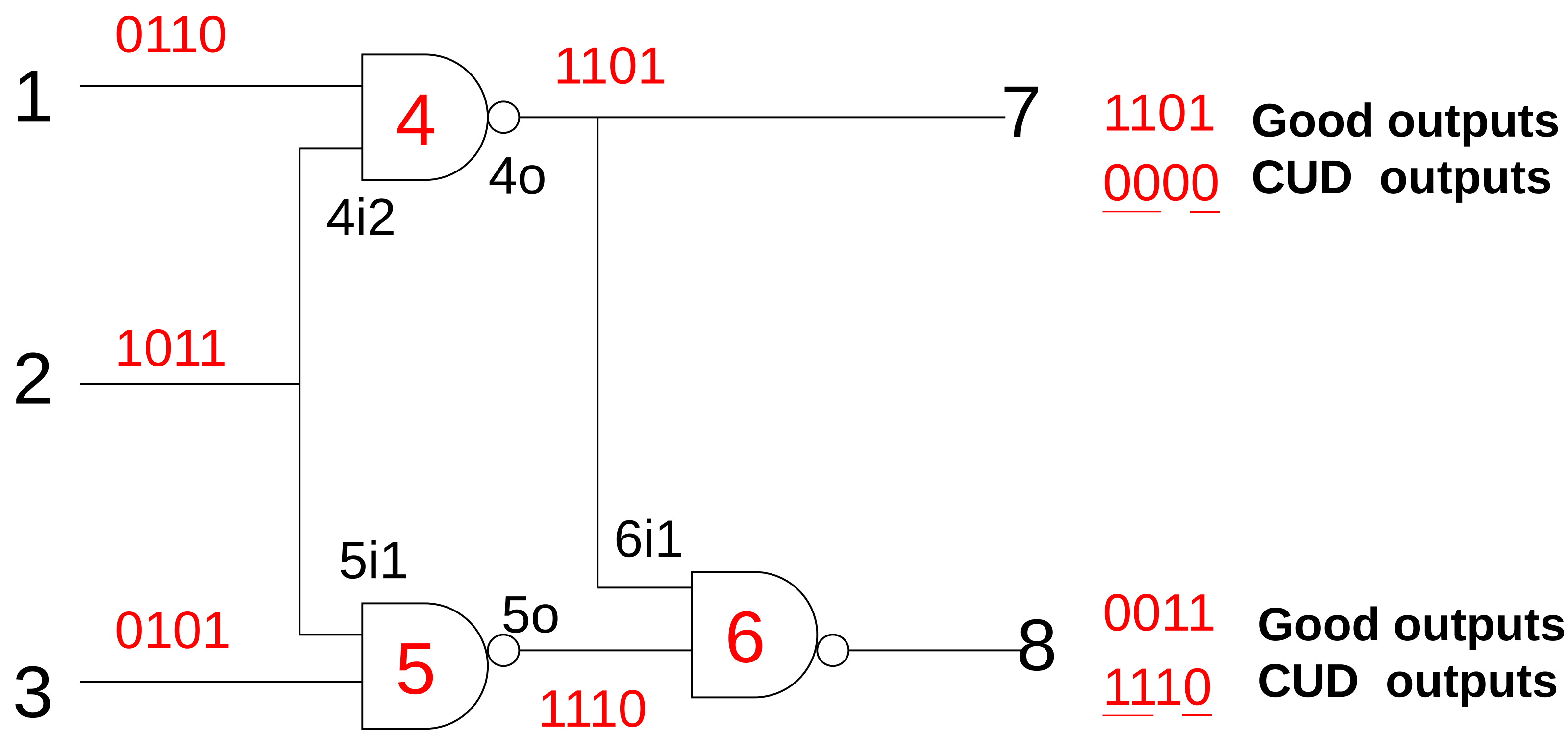
- Need very fast algorithm to select faults
- Idea: [Waicukauski 89]
  - Remove impossible candidate faults based on
    - structural or logic value
- Three steps:
  - 1. Structural backtracing
  - 2. Parity check
  - 3. Excitation condition check
- NOTE: This technique assumes SSF

Remove Impossible Candidate Faults Before Fault Simulation

Example CUD

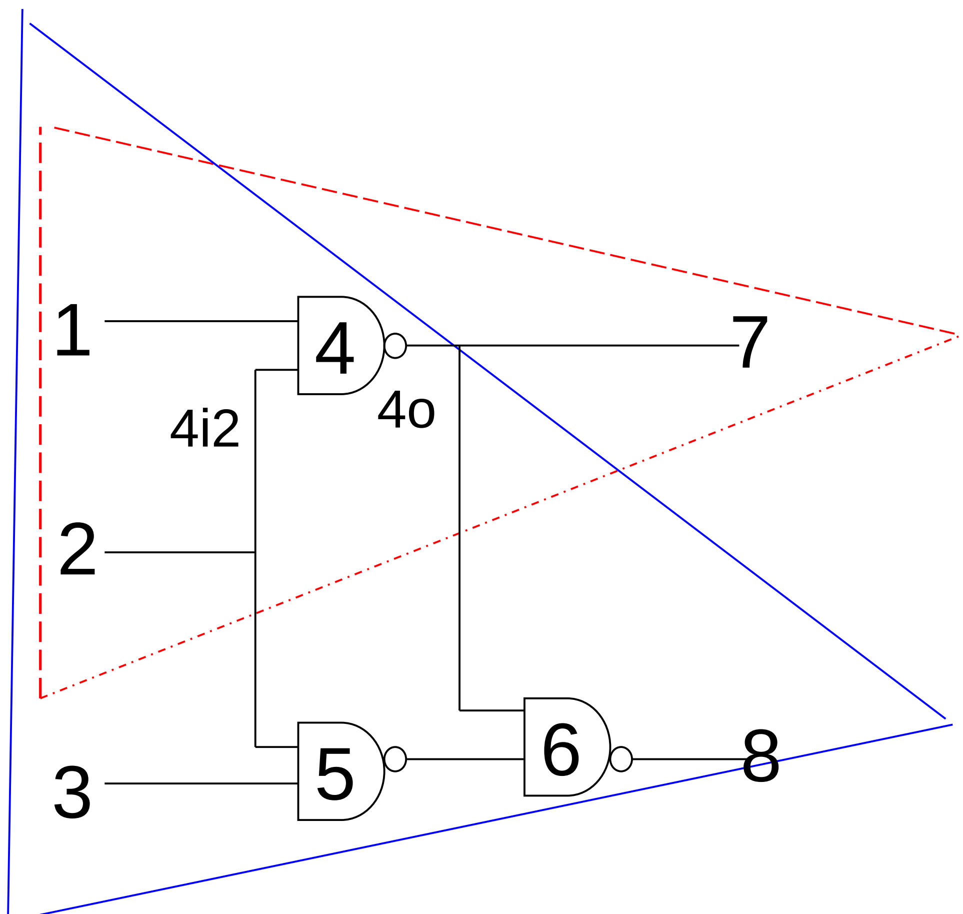
- **Given this CUD, test failures are**

	pattern1 7	8	pattern2 7	8	pattern3 7	8	pattern4 7	8
Test Failures	X	X	X	X			X	X



Step 1. Structural Backtracing

- ***Structural backtrace*** **from failing pins**
  - **True candidate fault must be in intersection of fanin cones**
- **Example: 14 faults ➔ 6 faults**
  - **1 sa1, 2 sa0, 2 sa1, 4o sa0, 4o sa1, 4i2 sa1 remains**

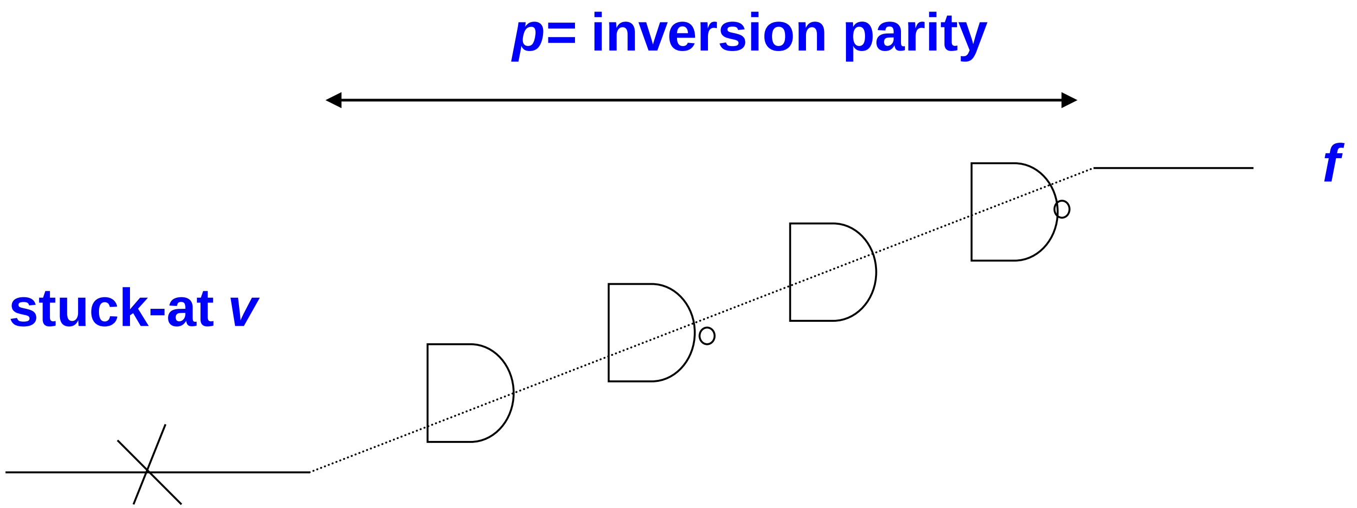


	original fault list
1	1 sa1
2	2 sa0
3	2 sa1
4	3 sa1
5	4o sa0
6	4o sa1; 4i2 sa0; 1sa0
7	4i2 sa1
8	5o sa1;3 sa0; 5i1 sa0
9	5i1 sa1
10	6i1 sa1
11	7 sa0
12	7 sa1
13	8 sa0
14	8 sa1; 6i1 sa0; 5osa0

Step 2. Parity Check

- For a **true candidate** stuck-at *v* fault, must satisfy

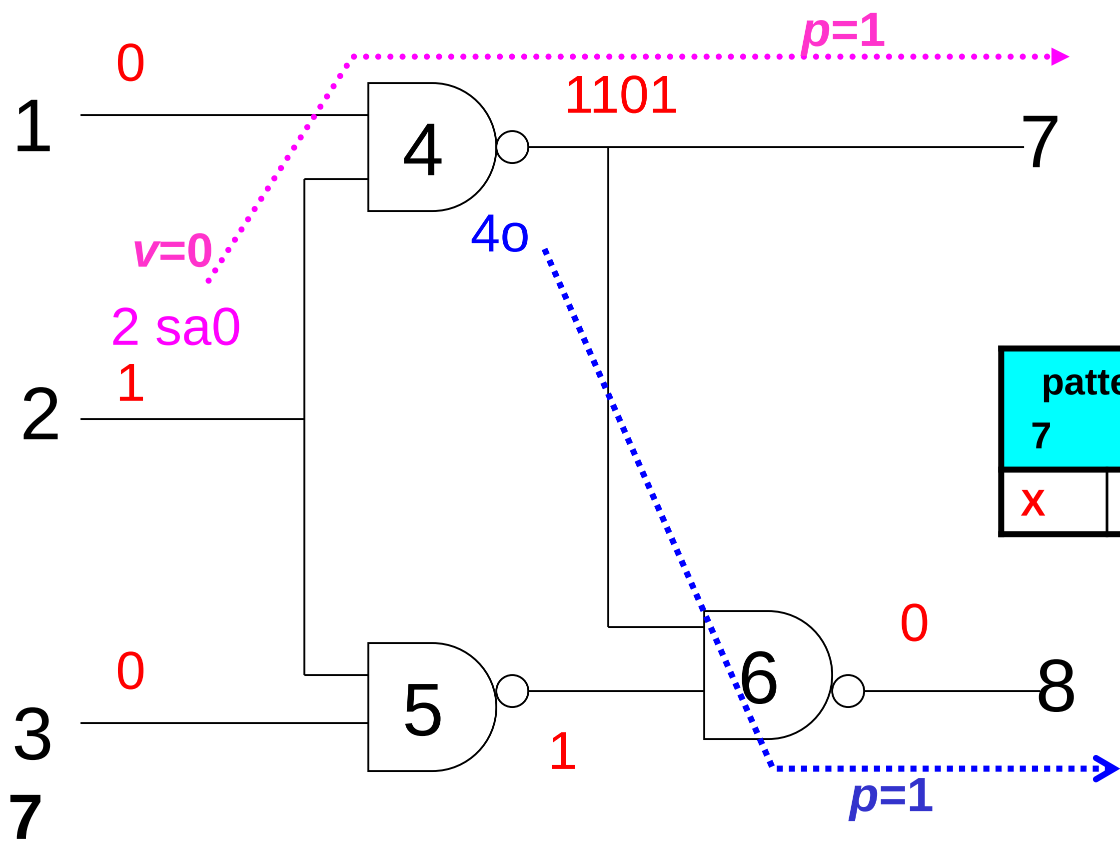
> $v \oplus p = f$
> *v* =stuck-at value
> *f* = CUD output value at a failing pin
> *p* = inversion parity on propagation path (1=odd, 0=even)



If a fault fails parity check, we can eliminate It.

Example

- For failing pattern 1
  - 2 sa0: *v*=0; *p*=1; *f*=0 parity check fails, fault eliminated
  - 4o sa1: *v*=1; *p*=1 *f*=1 parity check fails, fault eliminated
  - 4o sa0: *v*=0, *p*=1 *f*=1 parity check passes, fault remains
  - 6 faults ➔ 4 faults



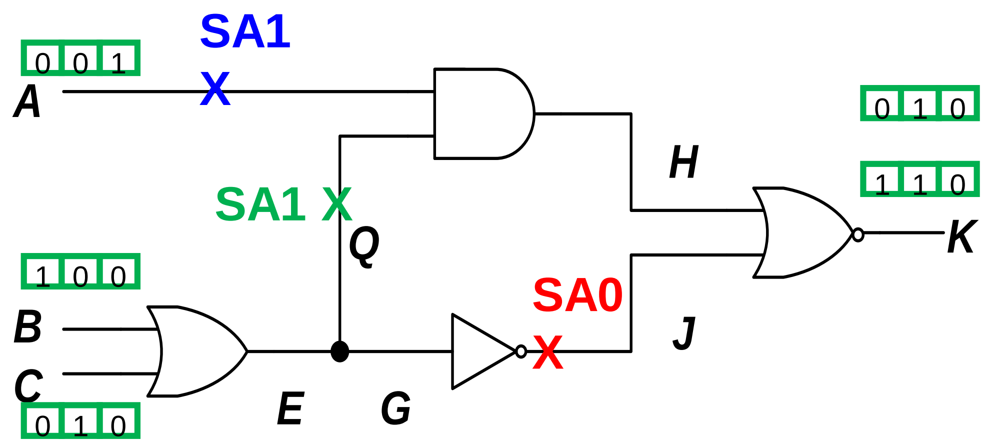
Test Failures

pattern1		pattern2		pattern3		pattern4	
7	8	7	8	7	8	7	8
x	x	x	x			x	

Quiz

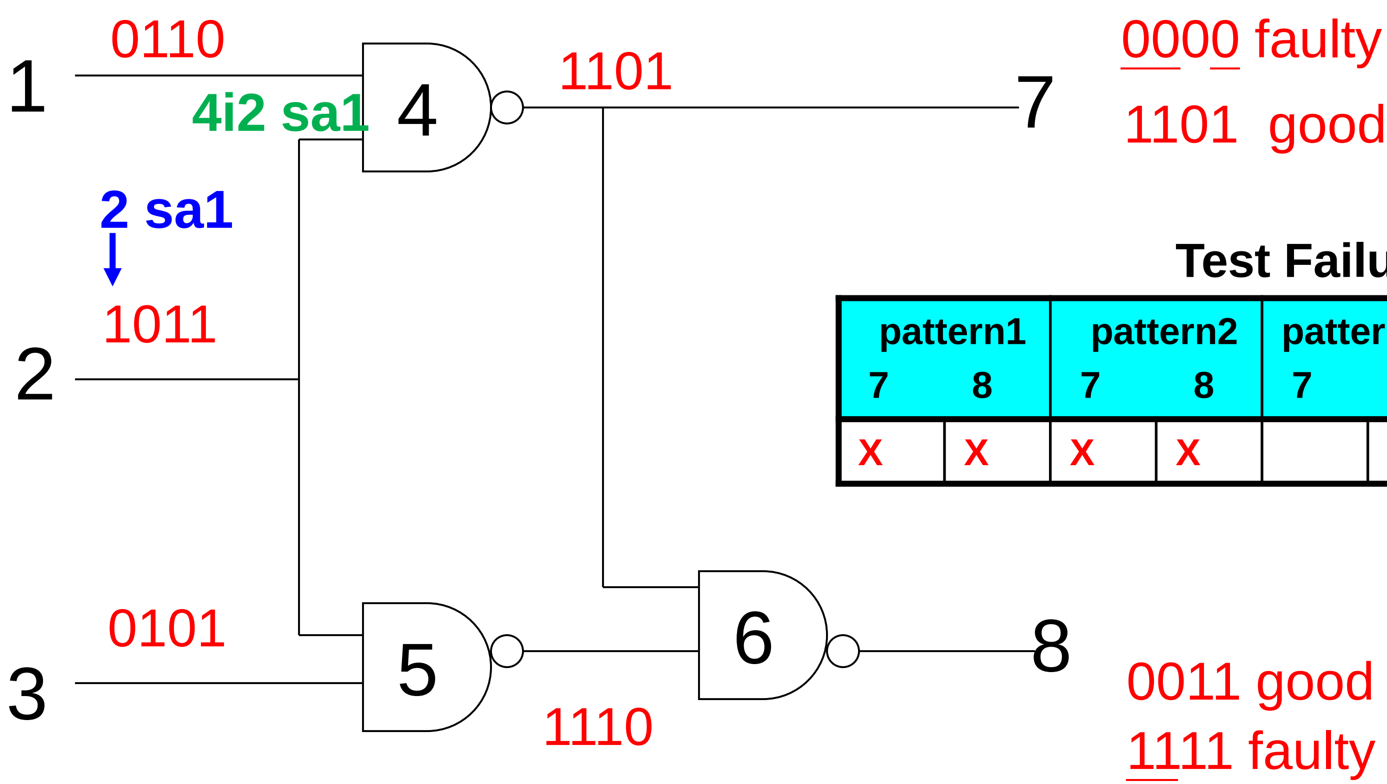
Q: Consider only three faults: *A SA1* , *J SA0, Q SA1*.
Apply 3 patterns: P_1= {010}, P_2= {001}, P_3= {100}.
Good outputs are {110}. CUD outputs are {010}.
Use parity check to see which faults should be eliminated.

	v	*p*	*f*	eliminate?
A SA1				
J SA0				
Q SA1				



Step 3. Excitation Condition Check

- For a true candidate *n* stuck-at *v* fault
  - *v* must differ from *n*'s good value in a failing pattern
- Example: 4 faults ➔ 2 faults
  - 2 sa1 and 4i2 sa1 eliminated (failing pattern 1)

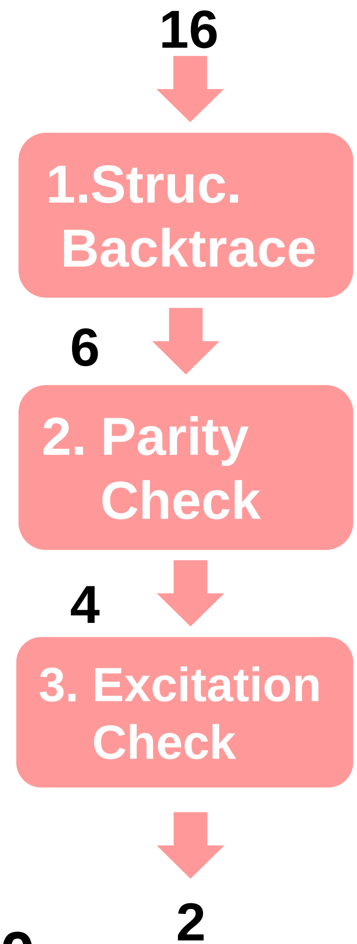


Test Failures

pattern1		pattern2		pattern3		pattern4	
7	8	7	8	7	8	7	8
x	x	x	x			x	

Partial Fault Dictionary

- Partial fault dictionary contains only 2 candidate faults
  - Much smaller than complete fault dictionary
- Finally 4o sa0 diagnosed most likely fault



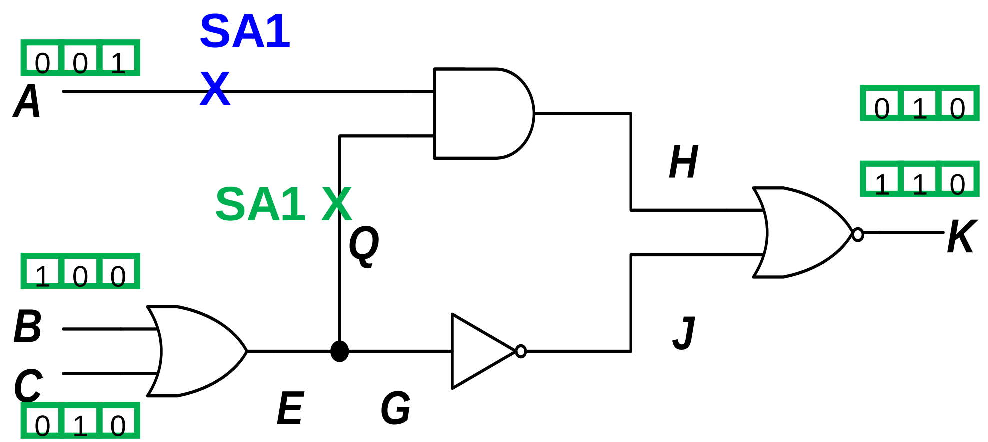
Partial fault dictionary

	faults	pattern1 7	pattern1 8	pattern2 7	pattern2 8	pattern3 7	pattern3 8	pattern4 7	pattern4 8
1	1 sa1	X	X					X	
5	4o sa0	X	X	X	X			X	
	Test Failures	X	X	X	X			X	

Quiz

Q: Now we have only two faults: *A SA1* , *Q SA1*.
Apply 3 patterns: P_1= {010}, P_2= {001}, P_3= {100}.
Good outputs are {110}. CUD outputs are {010}.
Use excitation check to see which faults should be eliminated.

	stuck value	good value	eliminate?
A SA1			
Q SA1			



Summary

- <u>Dynamic Cause-effect diagnosis</u>
  - First test CUD , then select few candidate faults to simulate
  - Three steps
    - 1. Structural backtracing
    - 2. Parity check
    - 3. Excitation condition check
  - Generate partial fault dictionary
    - Save storage space
  - Very useful in practice

FFT

- Q: Is step 2 still applicable when fanout branches reconverge?
  - If so, what is inversion parity?