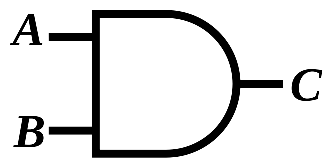
Fault Collapsing

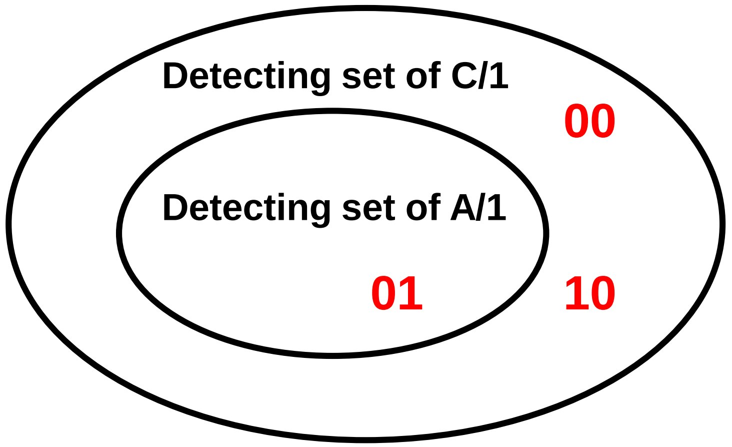
- Introduction
- Equivalence Fault Collapsing
  - Fanout-free circuits
  - Circuits with fanouts
- Dominance Fault Collapsing
  - **Fanout-free circuits**
  - **Circuits with fanouts**
- **Checkpoint Theorem (1976)**
- **Conclusion**

Fault Dominance

- *Detecting set* of fault *f* (T_f) = set of all test patterns that detect fault *f*
- Fault *f* ***dominates*** fault *g* if the detecting set of *f* contains that of *g*
  - $T_f \supseteq T_g$
- Example: C/1 fault dominates A/1 fault
  - C/1 detecting set {00, 01, 10} contains A/1 detecting set {01}
  - C/1 is ***dominating fault***; A/1 is ***dominated fault***



Input		Output						
A	*B*	good	*A/0*	*C/0*	*B/0*	*A/1*	*C/1*	*B/1*
0	0	0	0	0	0	0	1	0
0	1	0	0	0	0	1	1	0
1	0	0	0	0	0	0	1	1
1	1	1	0	0	0	1	1	1



Quiz

Q: Is dominance relation transitive?
if *f* dominates *g*, and g dominates *h*, then *f* dominates *h*?

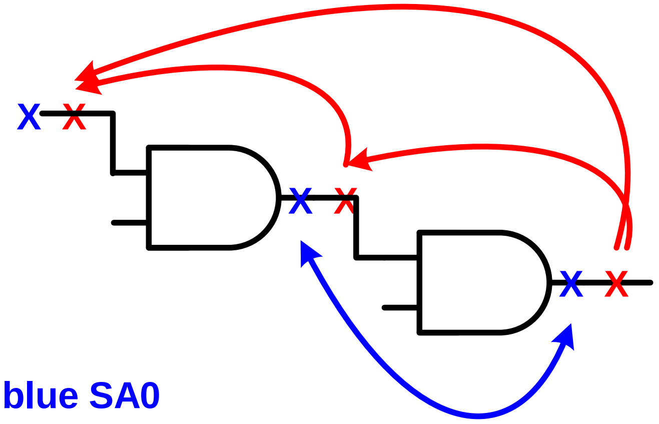
A:

Q: Is dominance relation symmetric?
if *f* dominates *g*, then *g* dominates *f*?

A:

f ↔ *g*: equivalence

f → *g*: dominance

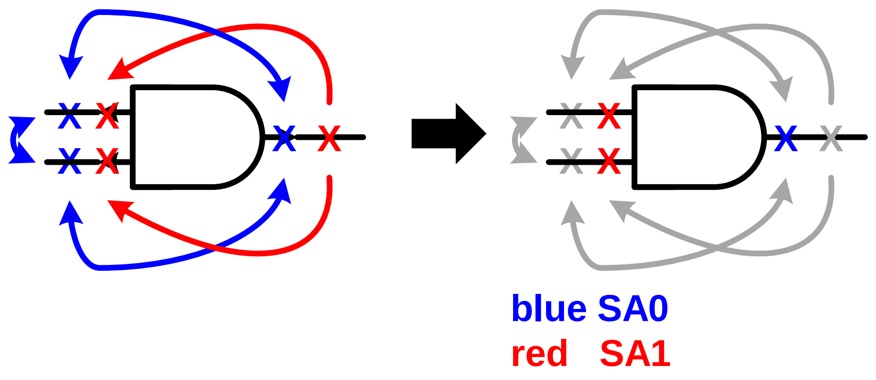


blue SA0
red SA1

Dominance Fault Collapsing, DFC

- DFC reduces fault list using fault dominance relation
  - If a test pattern detects a *dominated fault*, then it must detect its *dominating fault.* So *dominating faults can be removed* for ATPG
- Example:
  - C/1 fault can be removed. Similarly, A/0, B/0 also removed
  - Minimum {A/1, B/1, C/0} 3 faults remains after DFC

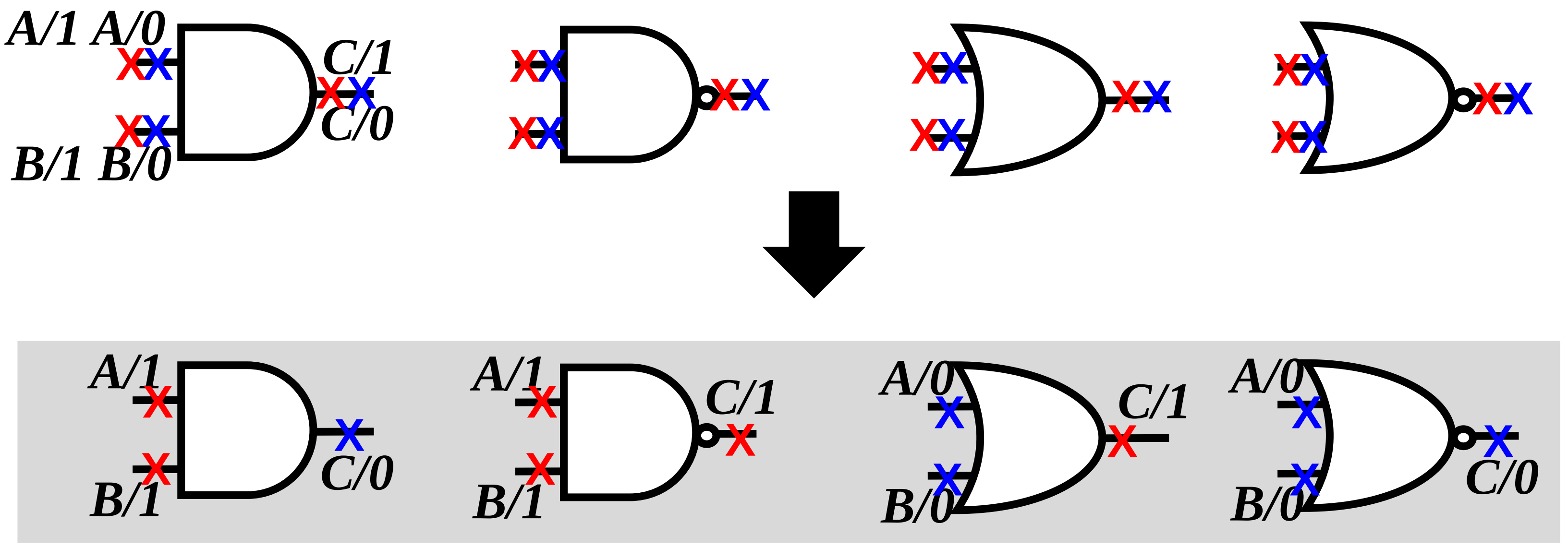
Input		Output						
A	*B*	Good	*A/0*	*C/0*	*B/0*	*A/1*	*C/1*	*B/1*
0	0	0	0	0	0	0	1	0
0	1	0	0	0	0	1	1	0
1	0	0	0	0	0	0	1	1
1	1	1	0	0	0	1	1	1



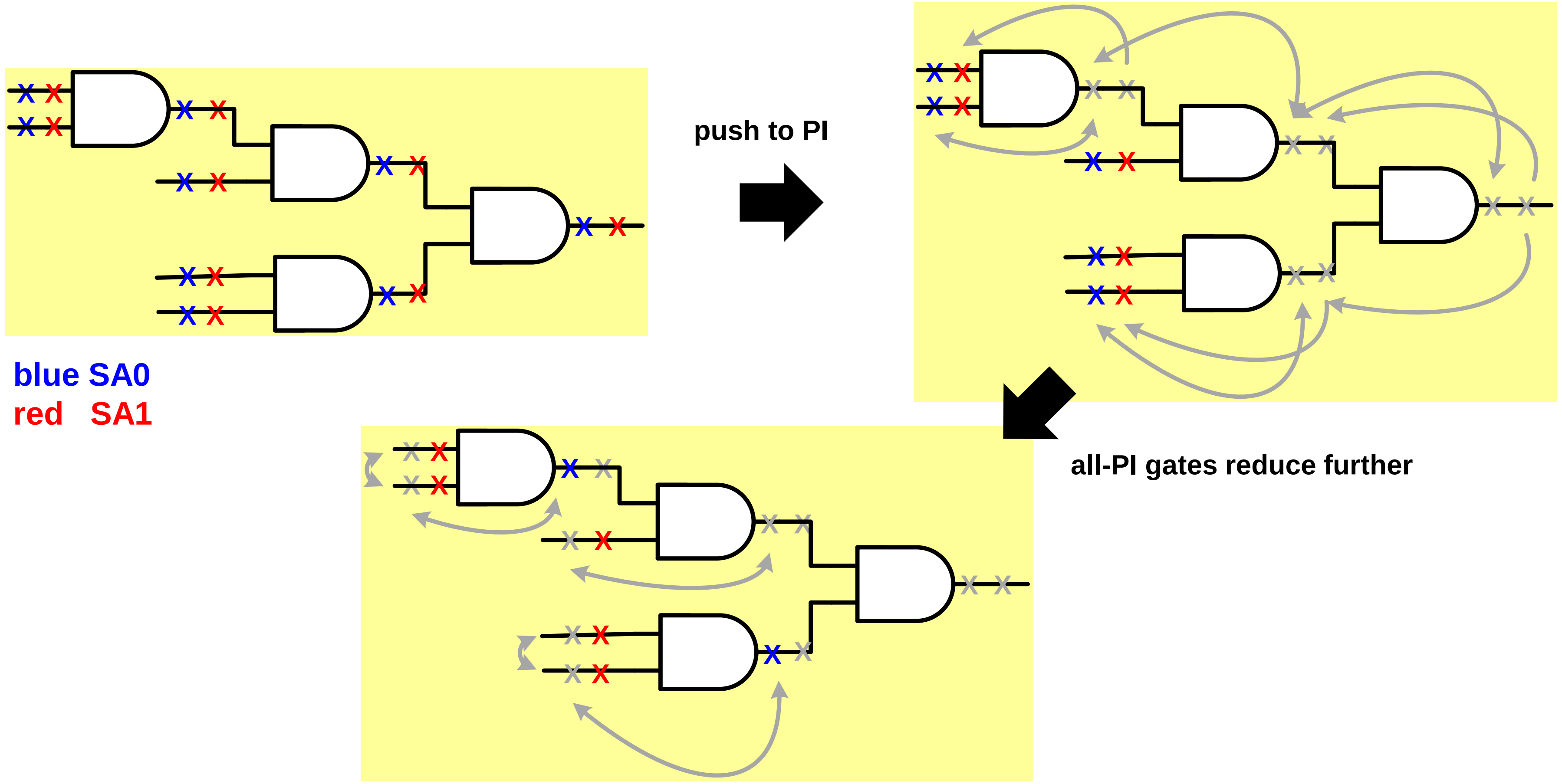
Minimum 3 faults after DFC

DFC on Elementary Gates

- **2-input AND/OR/NAND/NOR elementary gates**
  - **Originally 6 faults → 4 faults after EFC → 3 faults after DFC**
- **For *n*-input elementary gate,**
  - ***n*+1 stuck-at faults after DFC**



DFC on Fanout-free Circuits



- Two observations:
  - ① All gate input faults removed; except primary inputs
  - ② All gate output faults removed; except all-PI gates
- This is also applicable to other elementary gates. Why? FFT

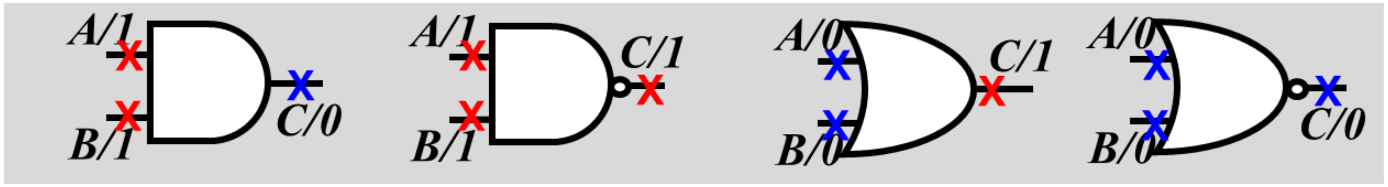
DFC Rules for Fanout-free Circuits

DFC Rules

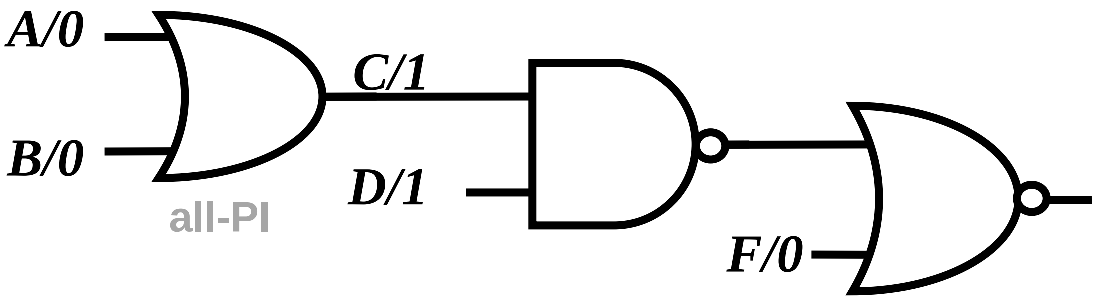
(1) one DFC collapsed fault for every primary input

(2) one DFC collapsed fault for each all-PI gate output*

*all-PI gate = gate inputs are *all* PI

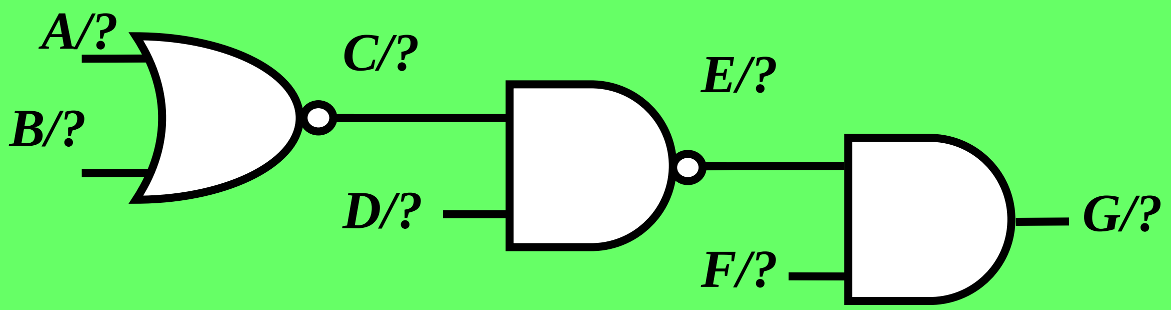


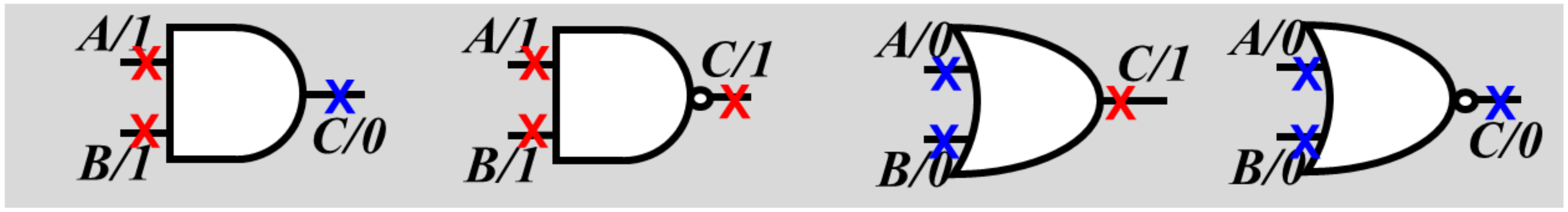
- **Example**
  - **Original 14 faults → 8 faults after EFC → 5 faults after DFC**



Quiz

Q: Please perform DFC on this fanout-free circuit. (Please label their stuck- values)





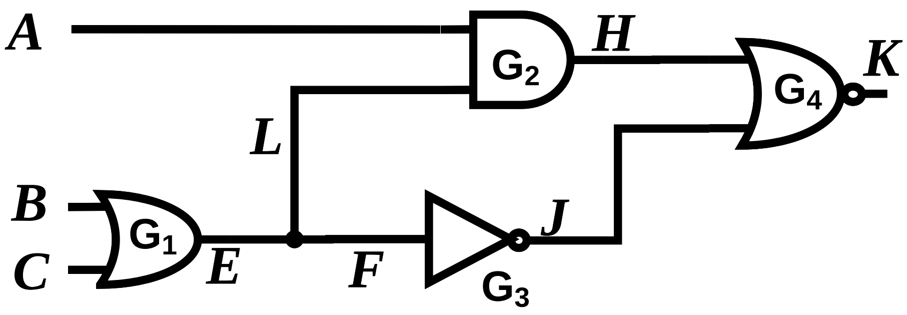
Fault Collapsing

- Introduction
- Equivalence Fault Collapsing
  - Fanout-free circuits
  - Fanout
- Dominance Fault Collapsing
  - Fanout-free circuits
  - <u>Circuits with fanout</u>
- Checkpoint Theorem
- Conclusion

Fanout Stem and Branches

- Fanout branch faults do NOT always dominate fanout stem faults
- Example: *F/1* fault dominates *E/1*. but *L/1* fault does not dominate *E/1*
- *Stem analysis* of fanout faults is time consuming

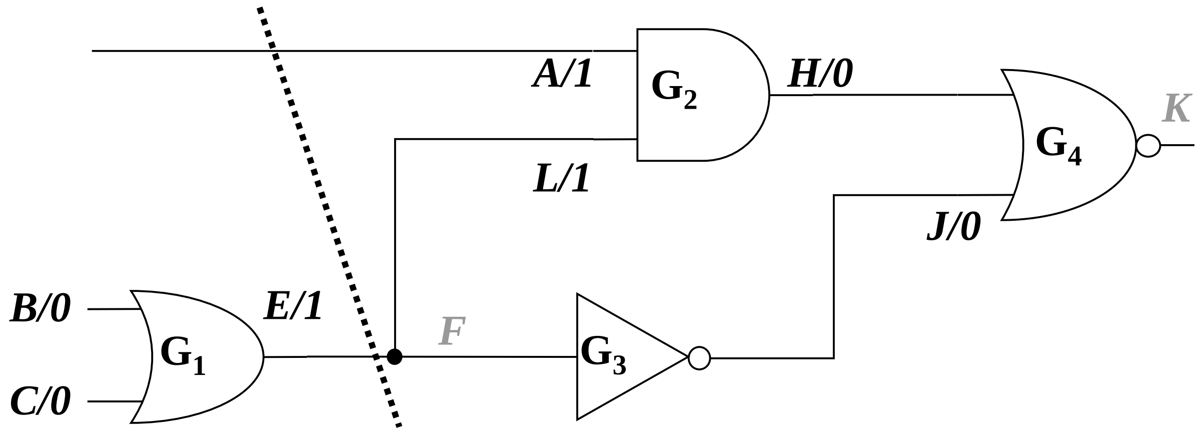
Input			Output						
A	*B*	*C*	good	*E/0*	*F/0*	*L/0*	*E/1*	*F/1*	*L/1*
0	0	0	0	0	0	0	$\underline{1}$	$\underline{1}$	0
0	0	1	1	$\underline{0}$	$\underline{0}$	1	1	1	1
0	1	0	1	$\underline{0}$	$\underline{0}$	1	1	1	1
0	1	1	1	$\underline{0}$	$\underline{0}$	1	1	1	1
1	0	0	0	0	0	0	0	$\underline{1}$	0
1	0	1	0	0	0	$\underline{1}$	0	0	0
1	1	0	0	0	0	$\underline{1}$	0	0	0
1	1	1	0	0	0	$\underline{1}$	0	0	0



Hard to Analyze Dominance between Stem and Branches

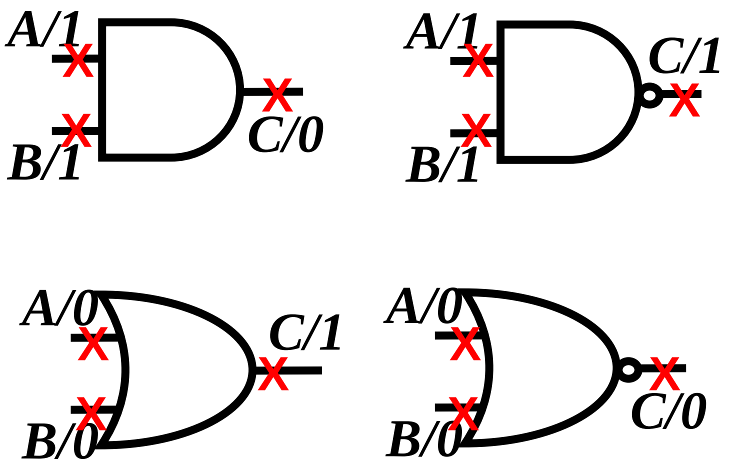
How to Handle Circuits w/ Fanouts?

- Approximate solution: Partition circuit into fanout-free subcircuits
  - DFC each subcircuit independently
- Example:
  - *J* is consider input because inverter G_3 is ignored
  - Original 18 faults, → after EFC 10 faults → after DFC 7 faults
  - NOTE: this solution is NOT optimal
    - *J/0* is equivalent to *F/1*,which dominates *E/*1



Simple_DFC Algorithm

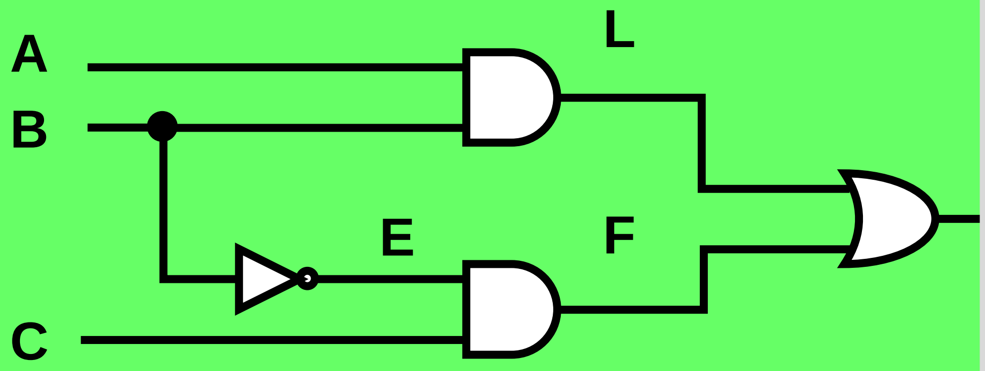
- **Linear time**
- **Does NOT guarantee minimum DFC**
  - **No stem analysis**



```
Simple_DFC (N)   /*N is a netlist*/
 0.  fault_list = {};
 1.  foreach gate or PI or PO g in N
 2.     if ((g is PI and fanout stem) || (g is PO and fanout branch)) then
 3.         fault_list = fault_list ∪ g output stuck-at 0 and 1;
 4.     else if (g is gate) then
 5.         foreach gate input i of gate g
 6.             h = backtrace inverters starting from i;
 7.             if (h is PI or fanout branch) then  /* rule #1 */
 8.                 if (gate g is AND) || (gate g is NAND) then
 9.                     fault_list = fault_list ∪ i stuck-at 1;
10.                 else if (gate g is OR) || (gate g is NOR)
11.                     fault_list = fault_list ∪ i stuck-at 0;
12.                 end if
13.             end if
14.         end foreach
15.         if (every gate input of g has a fault) then   /* rule #2 */
16.             if (gate g is AND) || (gate g is NOR) then
17.                 fault_list = fault_list ∪ g output stuck-at 0;
18.             else if (gate g is OR) || (gate g is NAND) then
19.                 fault_list = fault_list ∪ g output stuck-at 1;
20.             end if
21.         end if
22.     end if
23.  end foreach
24.  return (fault_list );
```

Quiz

Q: Please perform DFC on this circuit.
(Please label their stuck- values)



```
Simple_DFC ($N$)   /*$N$ is a netlist*/
 0. fault_list = {};
 1. foreach gate or PI or PO $g$ in $N$
 2.   if (($g$ is PI and fanout stem) || ($g$ is PO and fanout branch)) then
 3.     fault_list = fault_list $\cup$ $g$ output stuck-at 0 and 1;
 4.   else if ($g$ is gate) then
 5.     foreach gate input $i$ of gate $g$
 6.       $h$ = backtrace inverters starting from $i$;
 7.       if ($h$ is PI or fanout branch) then   /* rule #1 */
 8.         if (gate $g$ is AND) || (gate $g$ is NAND) then
 9.           fault_list = fault_list $\cup$ $i$ stuck-at 1;
10.         else if (gate $g$ is OR) || (gate $g$ is NOR)
11.           fault_list = fault_list $\cup$ $i$ stuck-at 0;
12.         end if
13.       end if
14.     end foreach
15.     if (every gate input of $g$ has a fault) then   /* rule #2 */
16.       if (gate $g$ is AND) || (gate $g$ is NOR) then
17.         fault_list = fault_list $\cup$ $g$ output stuck-at 0;
18.       else if (gate $g$ is OR) || (gate $g$ is NAND) then
19.         fault_list = fault_list $\cup$ $g$ output stuck-at 1;
20.       end if
21.     end if
22.   end if
23. end foreach
24. return (fault_list );
```

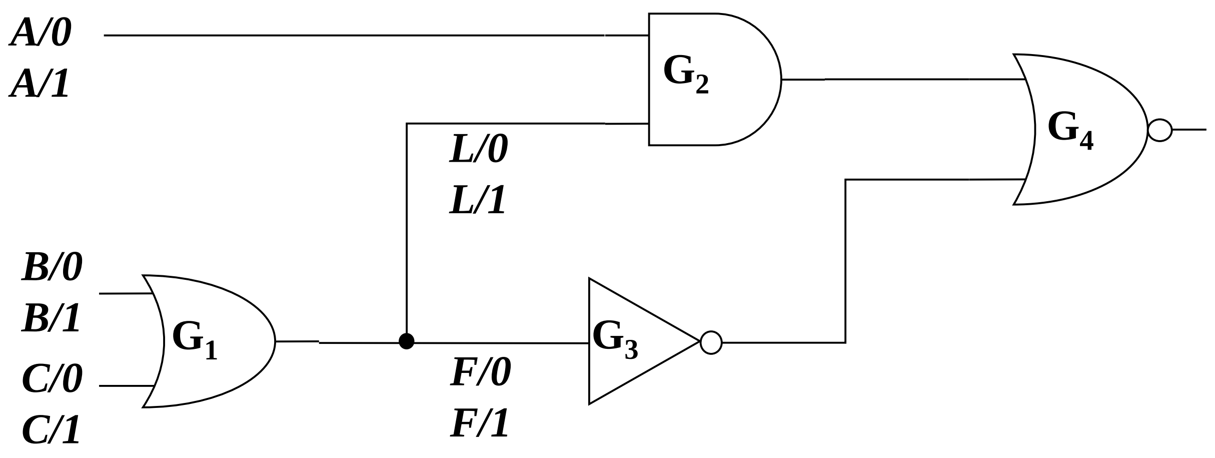
Fault Collapsing

- Introduction
- Equivalence Fault Collapsing
  - Fanout-free circuits
  - Fanout
- Dominance Fault Collapsing
  - Fanout-free circuits
  - Circuits with fanout
- <u>**Checkpoint Theorem**</u>
- Conclusion

Checkpoint Theorem [Breuer 76]

If a test detects all SSF on all checkpoints, this test detects all SSF in the circuit.

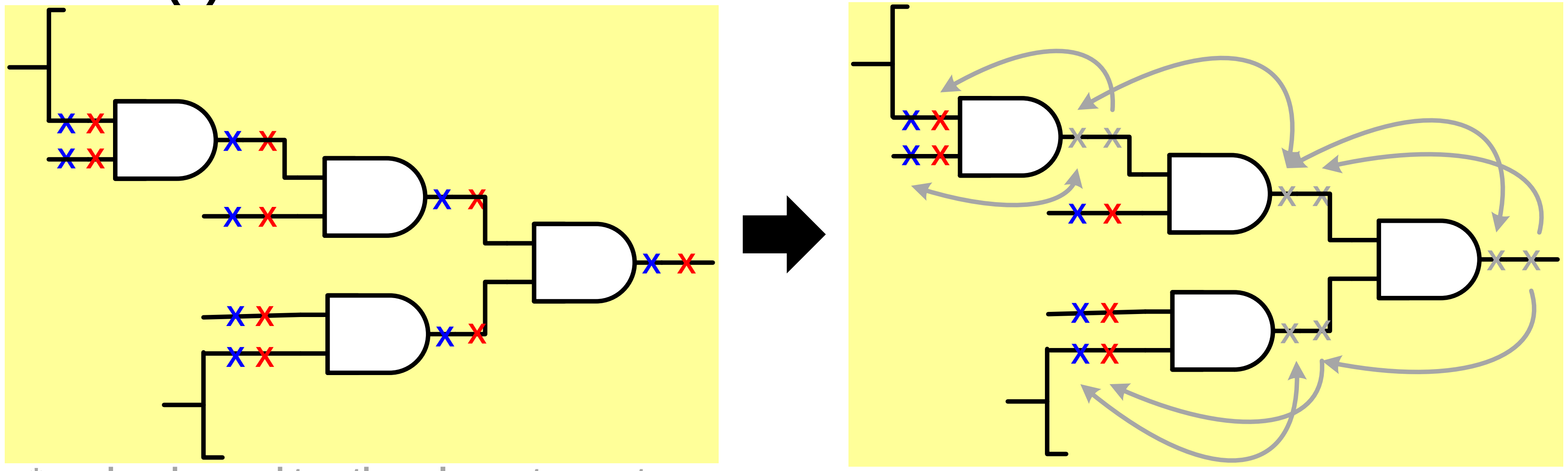
- Checkpoints = primary inputs + fanout branches
- Example:
  - 10 faults on checkpoints
  - Originally 18 faults, after EFC 10 faults, after DFC 7 faults



$|CHKPT| \geq |EFC| \geq |DFC|$

Proof

- Partition circuit into fanout-free subcircuits
- Use dominance relationship to remove faults from right to left until
  - (1) PI
  - (2) Fanout branch



*can be changed to other elementary gates

Chkpt are Simple Alternative to EFC/DFC

Fault Collapsing

- Introduction
- Equivalence Fault Collapsing
  - Fanout-free circuits
  - Fanout
- Dominance Fault Collapsing
  - Fanout-free circuits
  - Circuits with fanout
- Checkpoint Theorem
- **Conclusion**

Example ATPG Report

	uncollapsed	Collapsed (typically EFC)
Total Faults	1,234	800
Detected faults	1,000	700
Untestable faults	230	98
Aborted faults	4	2
Fault Coverage	1,000/1,234	700/800

$$CollapseRatio = \frac{\text{number of } collapsed \text{ faults}}{\text{number of } uncollapsed \text{ faults}} \times 100\% = \frac{800}{1234} \approx 64.8\%$$

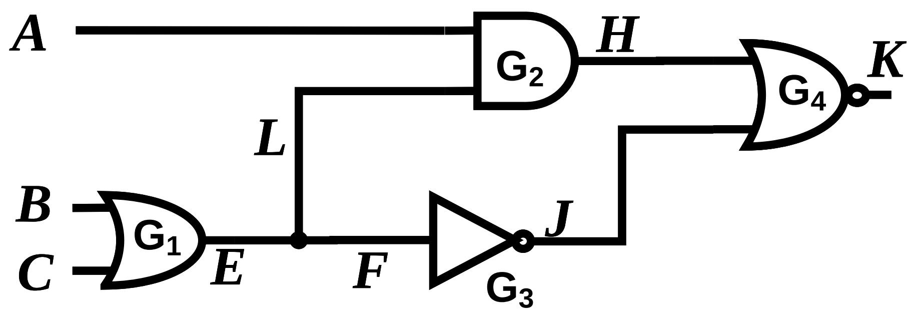
$$uncollapsed\ F.C. = \frac{\text{detected } uncollapsed \text{ faults}}{\text{total } uncollapsed \text{ faults}} \times 100\%$$

$$Collapsed\ F.C. = \frac{\text{detected } collapsed \text{ faults}}{\text{total } collapsed \text{ faults}} \times 100\%$$

Why DFC not Used in Practice?

- Because DFC fault coverage is *pessimistic*. *A* test pattern can detect a dominating fault without detecting its dominated fault
- Example: ABC=100 does *NOT* detect dominated fault E/1, but it detects dominating fault F/1
  - since F/1 is removed, this is not counted in DFC coverage

Input			Output						
A	*B*	*C*	good	*E/0*	*F/0*	*L/0*	*E/1*	*F/1*	*L/1*
0	0	0	0	0	0	0	1	1	0
0	0	1	1	0	0	1	1	1	1
0	1	0	1	0	0	1	1	1	1
0	1	1	1	0	0	1	1	1	1
1	0	0	0	0	0	0	0	1	0
1	0	1	0	0	0	1	0	0	0
1	1	0	0	0	0	1	0	0	0
1	1	1	0	0	0	1	0	0	0



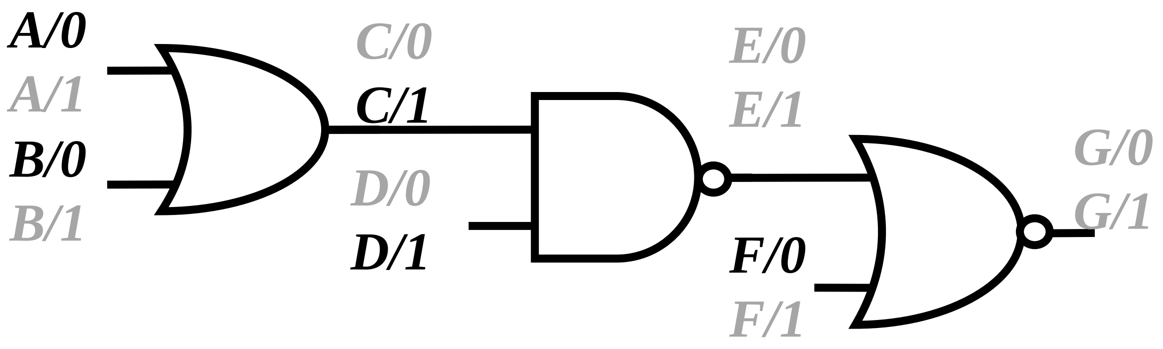
Conclusions

- Fault collapsing reduces size of fault list
  - Speed up ATPG, Shorten test length
- DFC Rules:
  - (1) one fault on PI (2) one fault on output of all-PI gate
- DFC is more aggressive the EFC
  - but DFC not often used because its FC is pessimistic

- Checkpoint Theorem:
  - Detecting all checkpoint faults is good enough
  - Checkpoints are (1) PI, (2) fanout branches

FFT

- Q1: We only show DFC rule on AND gate fanout-free circuits.
  - Does it also applicable to other elementary gates? Please show why the other faults can be reduced



- Q2: Can we DFC on XOR gate? How about EFC?

References

- [Breuer 76] M. A. Breuer and A. D. Friedman, Diagnosis & Reliable Design of Digital Systems. Woodland Hills, CA: Computer Science Press, 1976.