ECE 538

VLSI System Testing

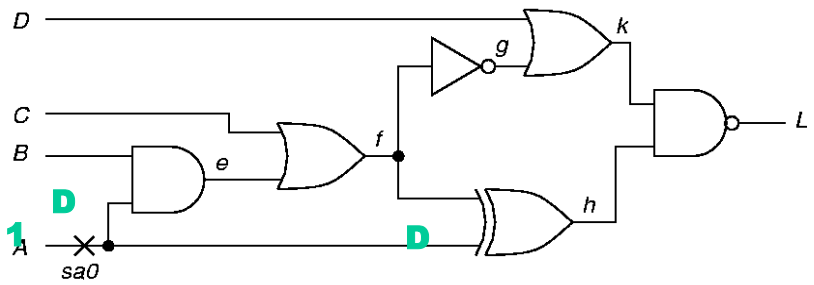
Krish Chakrabarty

Test Generation: 2

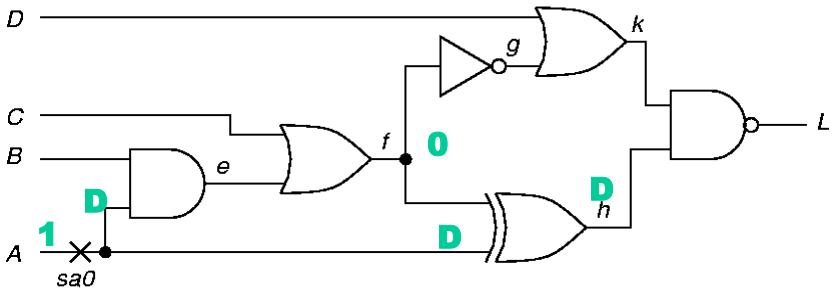
Outline

- Problem with D-Algorithm
- PODEM
- FAN
- Fault-independent ATPG
  - Critical path tracing
- Random test generation
- Redundancy identification

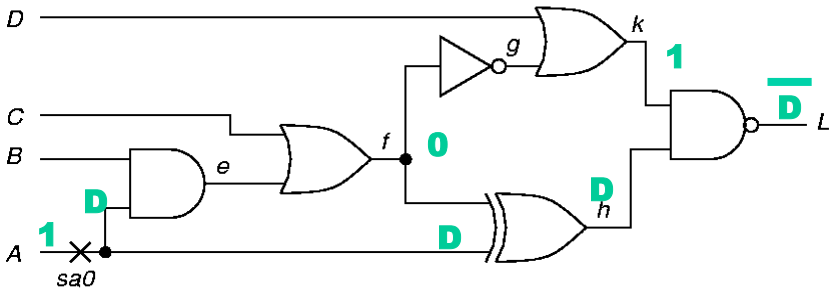
- Step 1 – *D-Drive* – Set *A = 1*



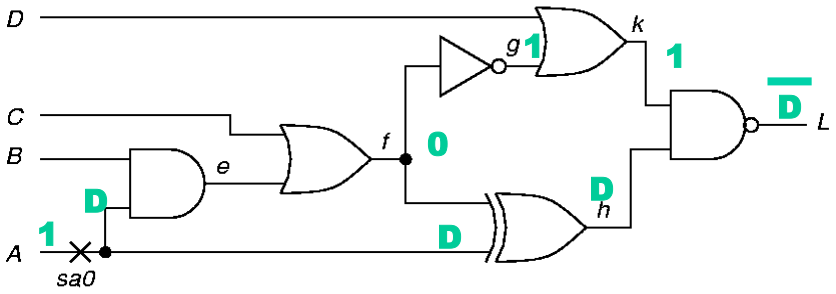
- Step 2 – *D-Drive* – Set *f* = 0



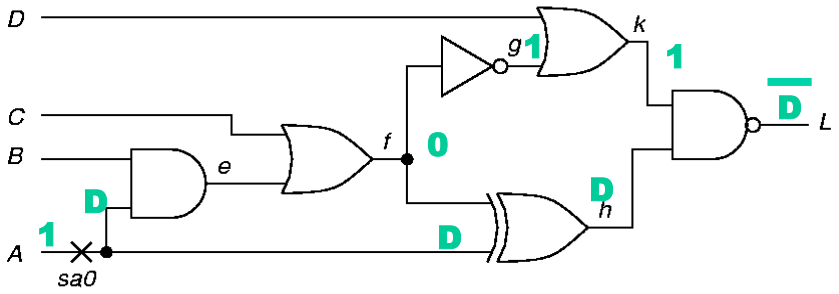
- Step 3 – *D-Drive* – Set $k = 1$



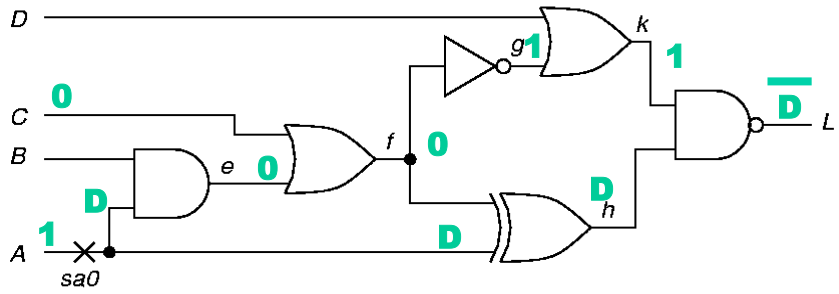
- Step 4 – *Consistency* – Set $g = 1$



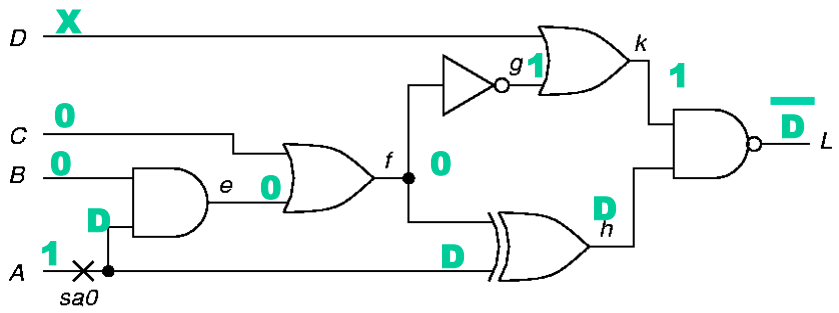
- Step 5 – *Consistency* – $f = 0$ Already set



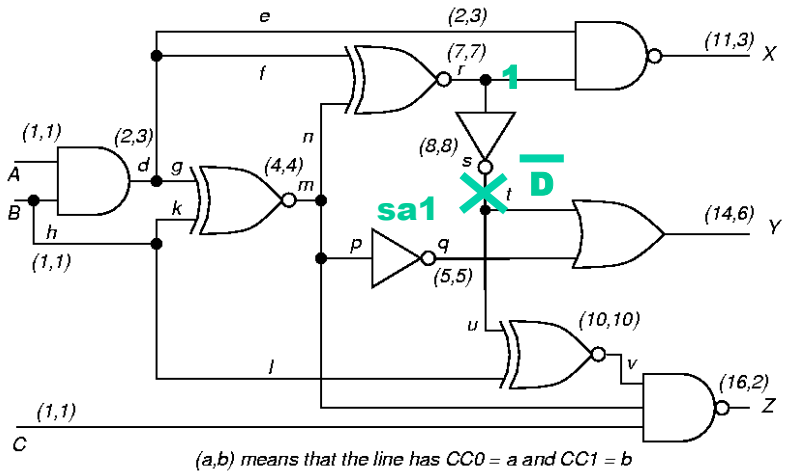
- Step 6 – *Consistency* – Set $c = 0$, Set $e = 0$



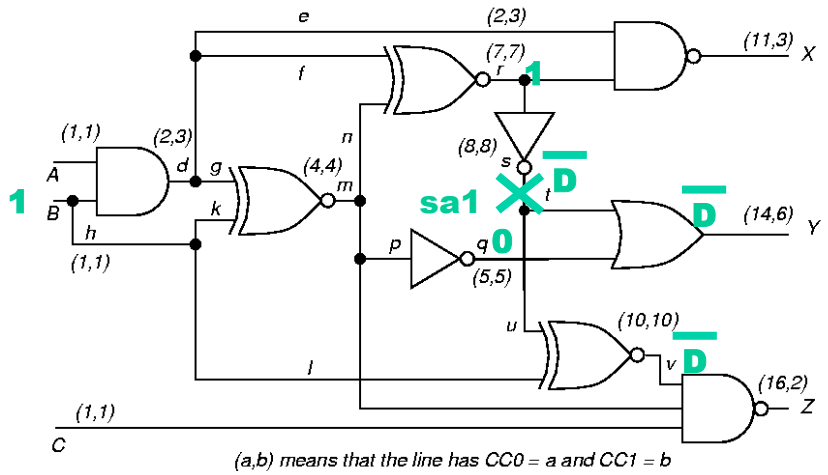
- Step 7 – *Consistency* – Set $B = 0$



Example 7.3 – Fault *s* sa1

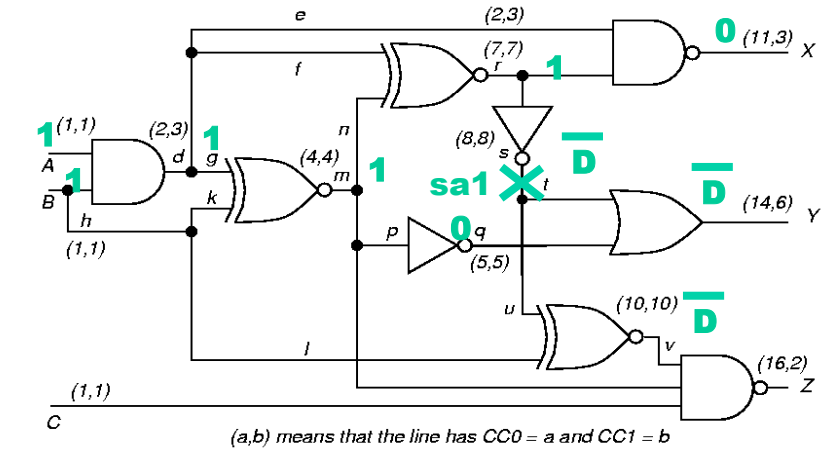


Example 7.3 – Step 2 *s* sa1



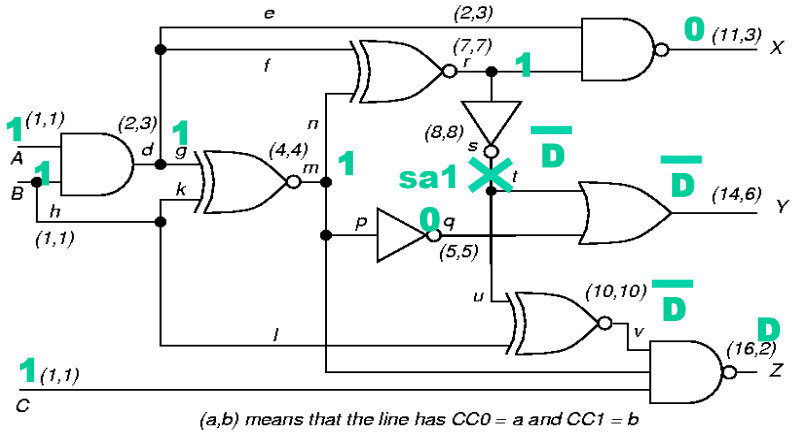
Example 7.3 – Step 2 *s* sa1

- Forward & Backward Implications

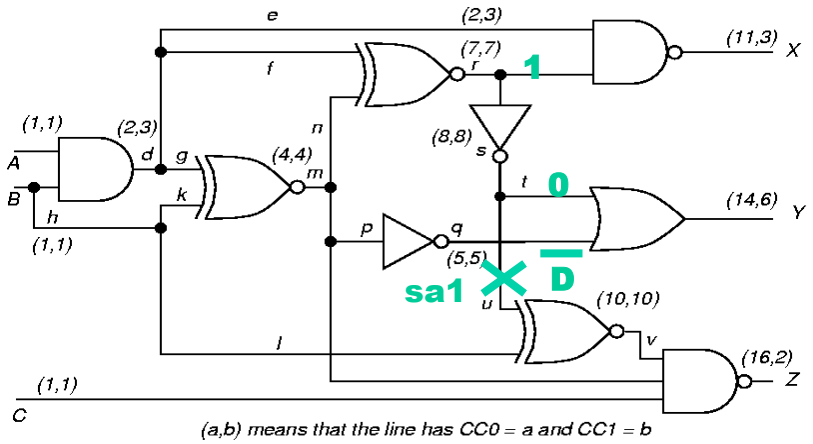


Example 7.3 – Step 3 *s* sa1

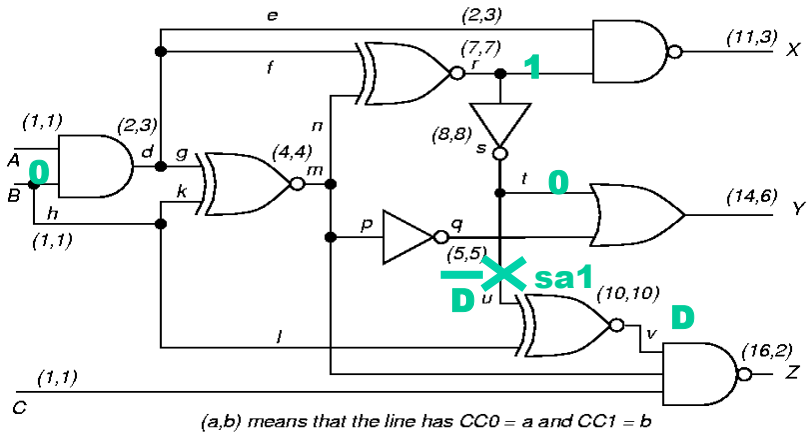
- Test found!



Example 7.3 – Fault *u* sa1

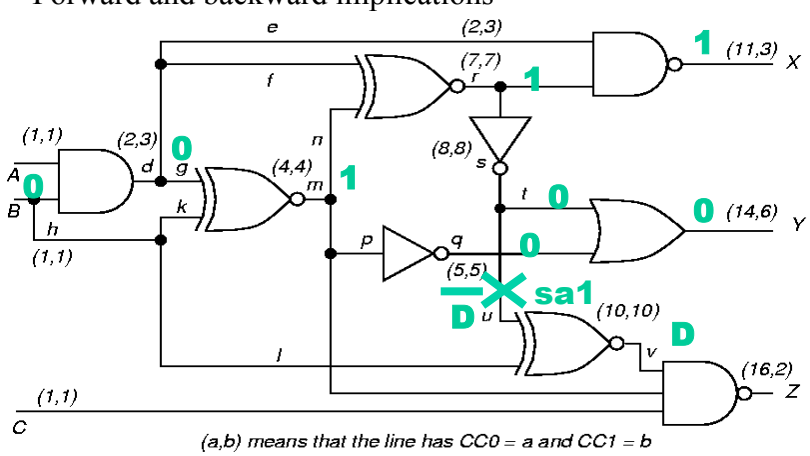


Example 7.3 – Step 2 *u* sa1



Example 7.3 – Step 2 *u* sa1

- Forward and backward implications

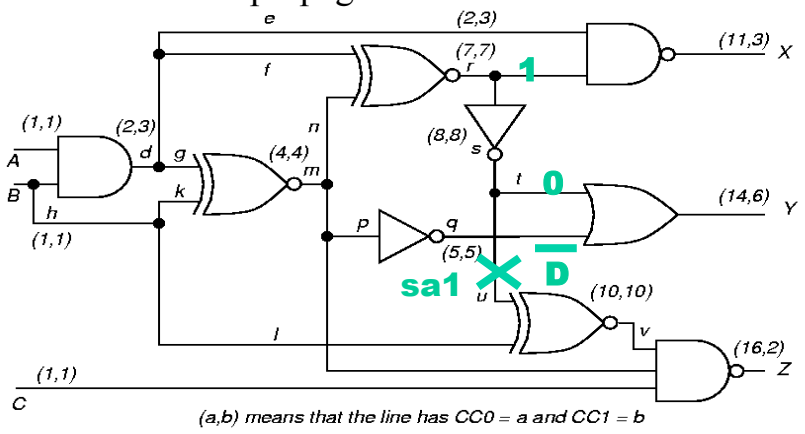


Inconsistent

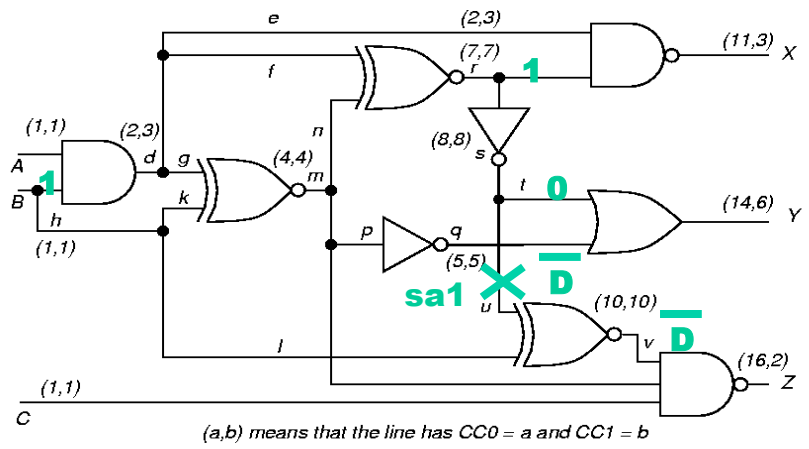
- $d = 0$ and $m = 1$ cannot justify $r = 1$ (equivalence)
  - Backtrack
  - Remove $B = 0$ assignment

Example 7.3 – Backtrack

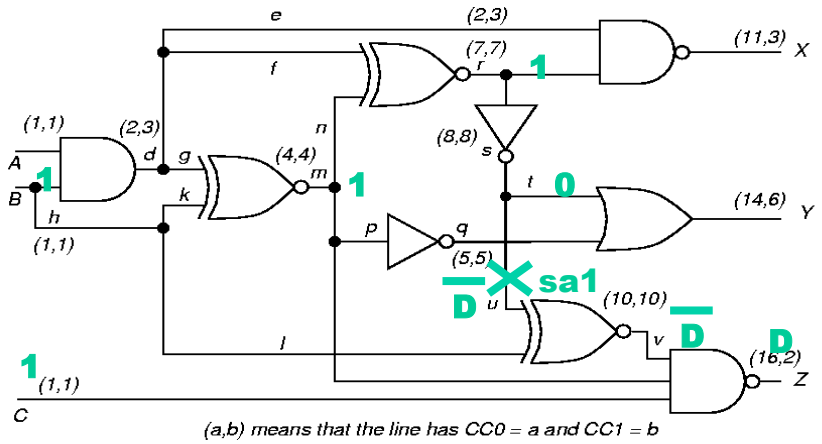
- Need alternate propagation



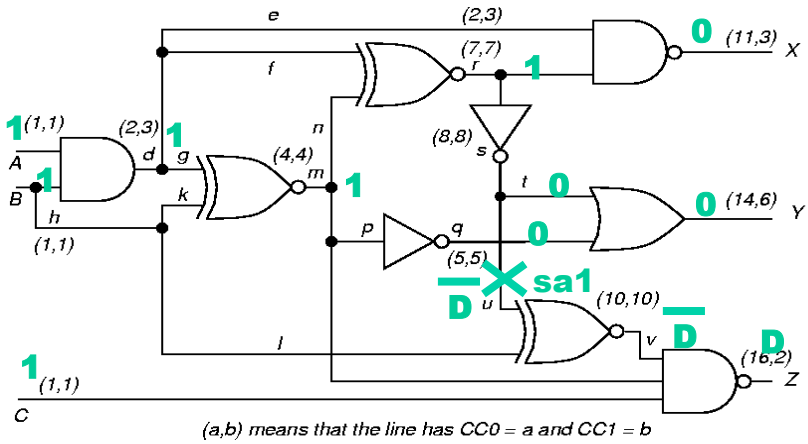
Example 7.3 – Step 3 *u* sa1



Example 7.3 – Step 4 *u* sa1

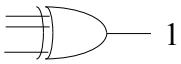


Example 7.3 – Step 4 *u* sa1



Problem with D-Algorithm

Excessive backtracking occurs in certain types of circuits

2^{n-1} justifying values  1

Causes "ripple effect" in many circuits, e.g. adders, parity circuits, error correcting circuits

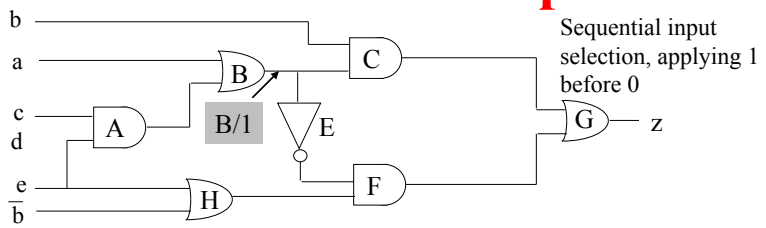
PODEM: "Path-Oriented Decision Making"

- *Similarity with D-algorithm*: circuit-based, fault-oriented
- *Difference*: Signal values explicitly assigned only at primary outputs, others computed by implication
- Justification not needed!
- Backtracking means reassigning primary inputs when contradiction occurs: "implicit enumeration"
- Simple "backtrace" heuristic used to select primary input

Branch and Bound Search

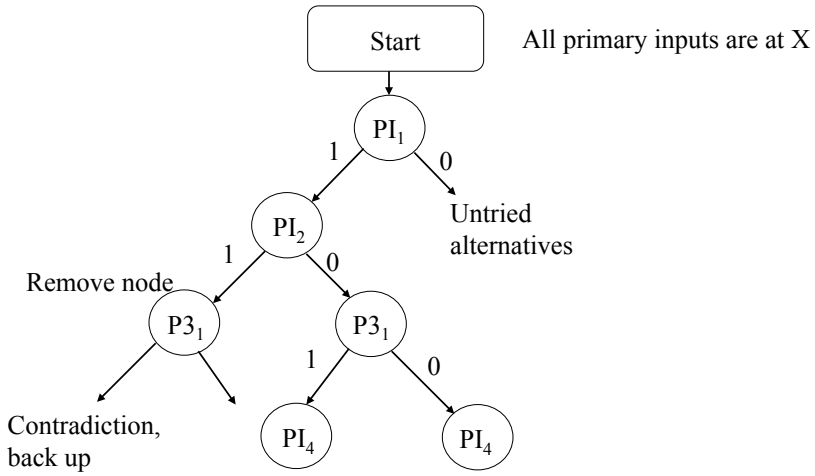
- Efficiently searches binary search tree
- *Branching* – At each tree level, selects which input variable to set to what value
- *Bounding* – Avoids exploring large tree portions by artificially restricting search decision choices
  - Complete exploration is impractical
  - Uses *heuristics*

PODEM Example



Decision	Implication	Comment
a = 1		Contradiction at fault Backtrack
a = 0		
b = 1		
c = 1		
e = 1	A = 1, B = 1	Contradiction, backtrack
e = 0	A = 0, B = $\overline{D}$, E = D, H = 0, F = 0, C = $\overline{D}$, z = $\overline{D}$	Test found!

PODEM Decision Tree



PODEM Steps

- Input Assignment
  - Unassigned PIs are selected and assigned new values systematically
  - All implications of each assignment are determined
  - If $D/\overline{D}$ is implied on a primary output, a test has been found; otherwise a new assignment or a new primary input line is selected

PODEM Steps

- Primary inputs selection:
  - INITIAL OBJECTIVE: A series of "initial" objectives of the form $IO_j = (l,v)$ are determined. The first IO_0 is to apply $v = D/\overline{D}$ to the fault site.
  - BACKTRACING: For each initial objective IO_j, a path is traced backwards through the circuit to a primary input via a series of "current" objectives
  - Current objectives are selected by heuristics

PODEM Procedures

```
Procedure Backtrace(k,v_k)
/* Map objective into PI assignment */
begin
 v = v_k;
  while k is a gate input
    begin
      i = inversion value of k;
      select an input (j) of k with value x;
      v = v ⊕i;
      k = j;
    end
 /* k is a PI */
  return (k,v);
end
```

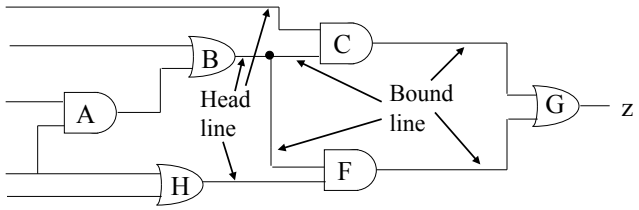
```
Procedure Objective()
begin
  /* the target is l/v */
 if (value of l is X) then return (l,v);
 select a gate (G) from the D-frontier;
 select an input (j) of G with value X;
 c = controlling value of G;
 return (j,c);
end
```


PODEM Procedures

```
PODEM()
begin
  if (error at PO) then return SUCCESS
  if (test not possible) then return FAILURE
 (k,v_k) = Objective();
 (j,v_j) = Backtrace(k,v_k);
 Imply(j,v_j);
if PODEM() = SUCCESS then return SUCCESS
/* reverse decision */
Imply(j,v_j);
if PODEM() = SUCCESS then return SUCCESS;
Imply(j,X);
return FAILURE
```

FAN: "Fanout-Oriented Test Generation"

- Two major extensions to PODEM
  - Backtracing may stop at internal lines
  - Multiple backtrace-procedures attempts to simultaneously satisfy a set of objectives
- Backtracing can stop at *head lines*

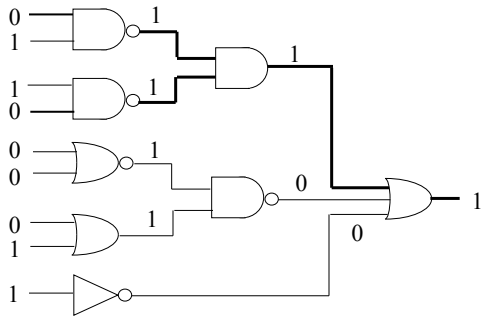


Selection Criteria

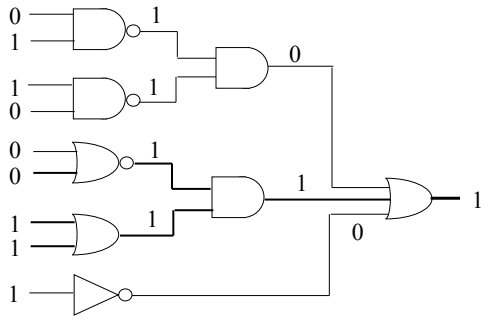
- Controllability (CC0 and CC1) and observability measures (CO)
  - Exact values can only be determined by exhaustive simulation
  - Estimates are useful for guiding test generation (more controllable ⇔ low values, more observable ⇔ low values)

Critical Path Test Generation

- Recursively determine critical paths



Critical Path Test Generation

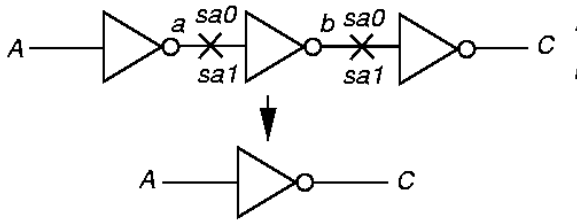


Redundancy Removal Using ATPG

- Redundancy identification
- Redundancy removal

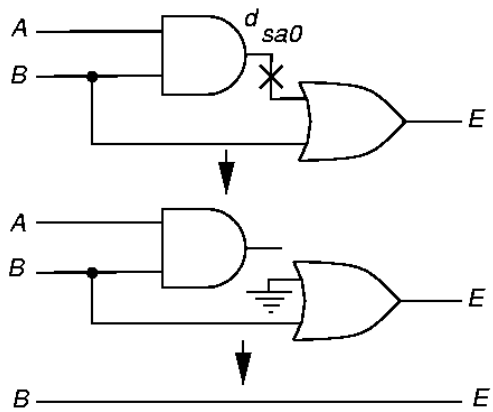
Irredundant Faults

- Combinational ATPG can find redundant (unnecessary) hardware

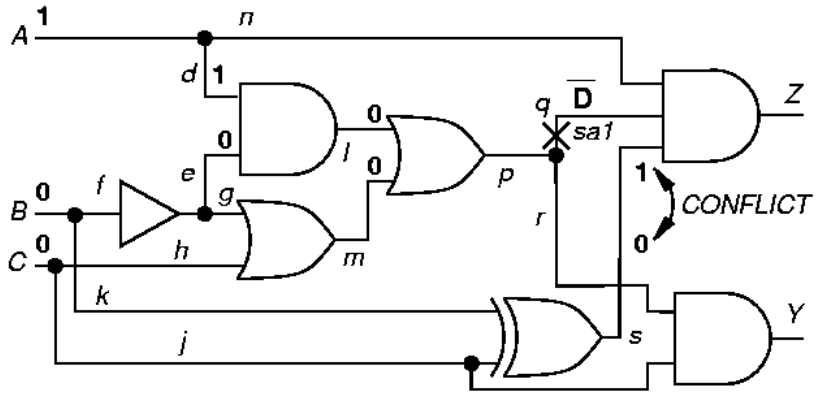


- Fault Test
  a sa1, *b* sa0 $A = 1$
  a sa0, *b* sa1 $A = 0$
- Therefore, these faults are not redundant

Redundant Hardware and Simplification

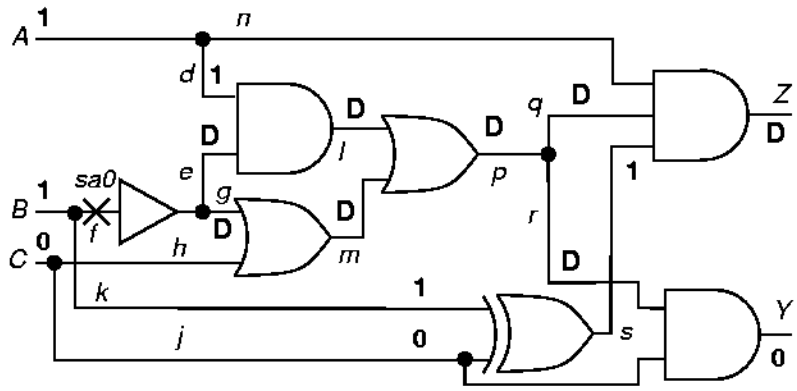


Redundant Fault Example



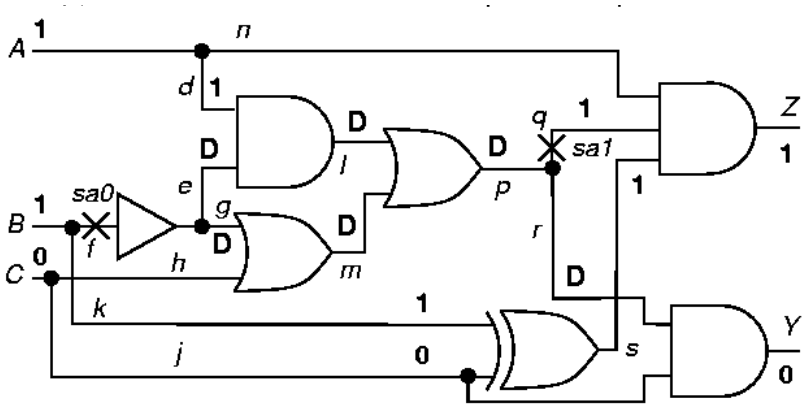
Multiple Fault Masking

- *f* sa0 tested when fault *q* sa1 not there



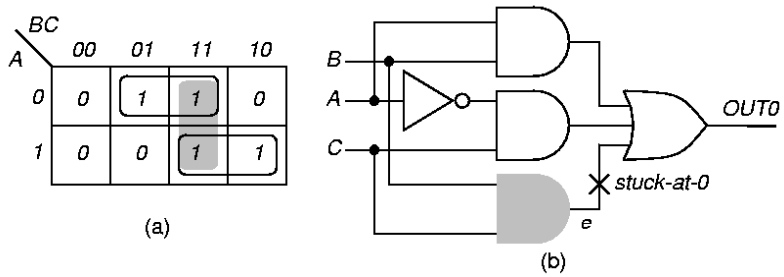
Multiple Fault Masking

- *f* sa0 masked when fault *q* sa1 also present



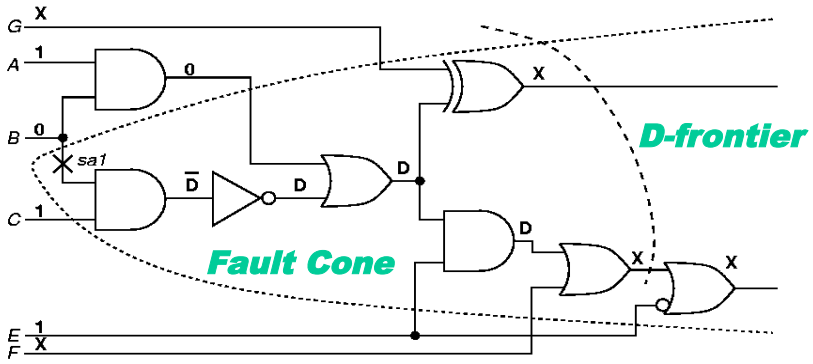
Intentional Redundant Implicant *BC*

- Eliminates hazards in circuit output



Fault Cone and D-Frontier

- *Fault Cone* -- Set of hardware affected by fault
- *D-frontier* – Set of gates closest to POs with fault effect(s) at input(s)



Redundancy Removal

Repeat until there are no more redundant faults:

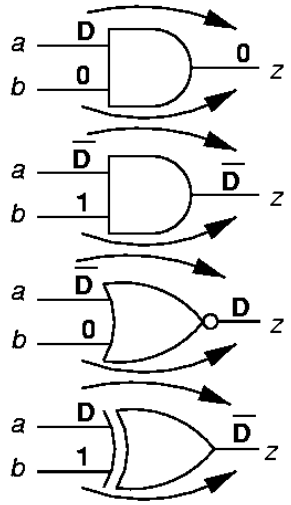
{

Use ATPG to find all redundant faults;

Remove all redundant faults with non-overlapping fault cone areas;

}

Forward Implication

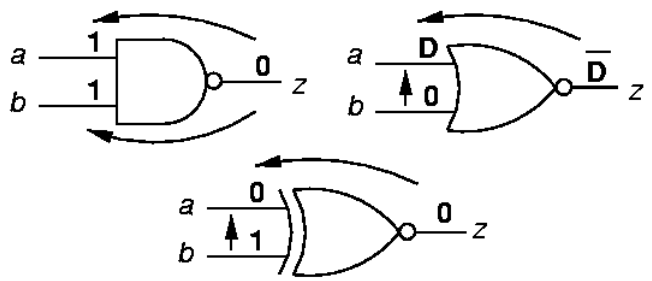


- Results in logic gate inputs that are significantly labeled so that output is uniquely determined
- AND gate forward implication table:

a \ b	0	1	X	D	$\overline{D}$
0	0	0	0	0	0
1	0	1	X	D	$\overline{D}$
X	0	X	X	X	X
D	0	D	X	D	0
$\overline{D}$	0	$\overline{D}$	X	0	$\overline{D}$

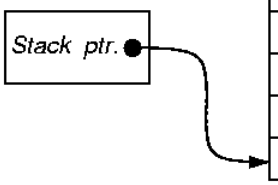
Backward Implication

- Unique determination of all gate inputs when the gate output and some of the inputs are given



Implication Stack

- Push-down stack. Records:
  - Each signal set in circuit by ATPG
  - Whether alternate signal value already tried
  - Portion of binary search tree already searched

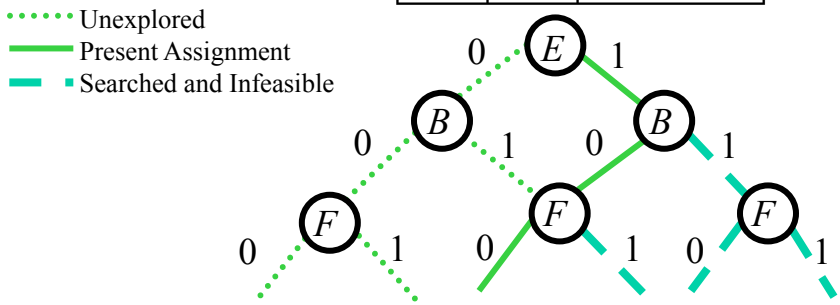


Signal	Value	Alternative tried
A	1	NO
C	1	NO
E	1	NO
B	0	YES

Implication Stack after Backtrack

Stack ptr.

Signal	Value	Alternative tried
E	1	NO
B	0	YES
F	0	YES



Implication Stack after Backtrack

Stack ptr.

Signal	Value	Alternative tried
E	1	NO
B	0	YES
F	0	YES