MIC05: Teste de Circuitos Integrados

Marcelo Lubaszewski

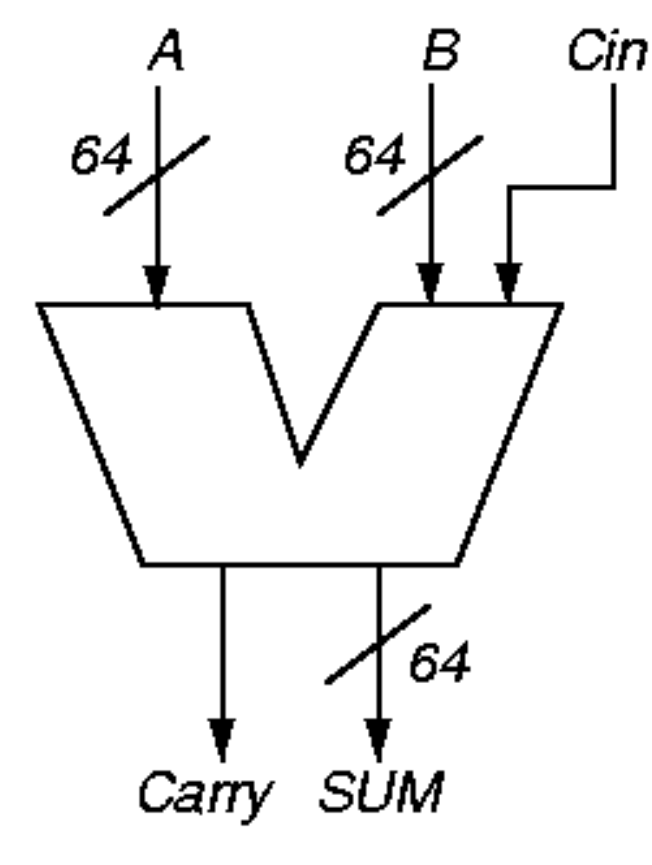
PGMICRO - UFRGS

2007/II

Lecture 5 – Test Pattern Generation

- Automatic test pattern generation
  - Structural vs. functional test
  - Definitions
  - Types of Algorithms
  - Summary

Functional vs. Structural ATPG



Functional ATPG

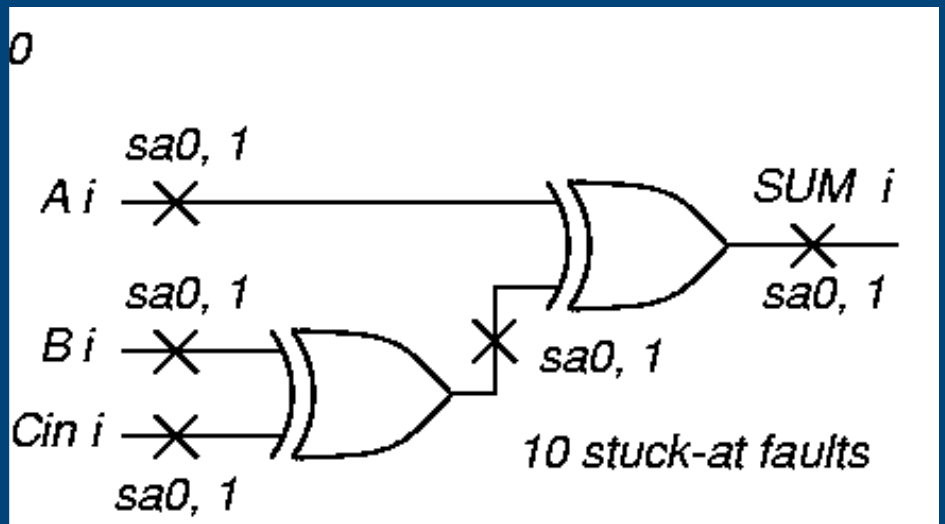
- generate complete set of tests for circuit input-output combinations
- 129 inputs, 65 outputs:

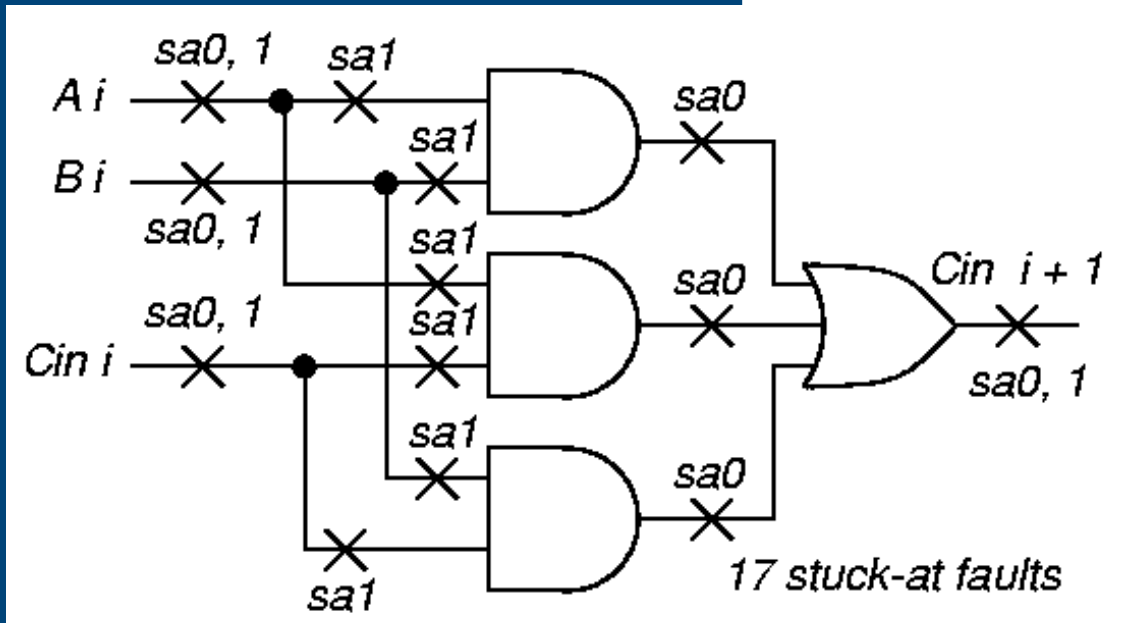
2^{129} = 680,564,733,841,876,926,926,749, 214,863,536,422,912

patterns

- Using 1 GHz ATE, would take 2.15×10^{22} years

Sum and Carry Circuits





Functional vs. Structural (Cont'd)

- Structural test:
  - No redundant adder hardware, 64 bit slices
  - Each with 27 faults (using fault equivalence)
  - At most 64 x 27 = 1728 faults (tests)
  - Takes 0.000001728 s on 1 GHz ATE

- Designer gives small set of functional tests – augment with structural tests to boost coverage to 98^{+} %

Definition of Automatic Test-Pattern Generator

- Operations on digital hardware:
  - Inject fault into circuit modeled in computer
  - Use various ways to activate and propagate fault effect through hardware to circuit output
  - Output flips from expected to faulty signal
- Test generation cost
  - fault-dependent or not
- Quality of generated test
  - fault coverage (fault simulation)
- Test application cost
  - test time, memory requirements

TG Types

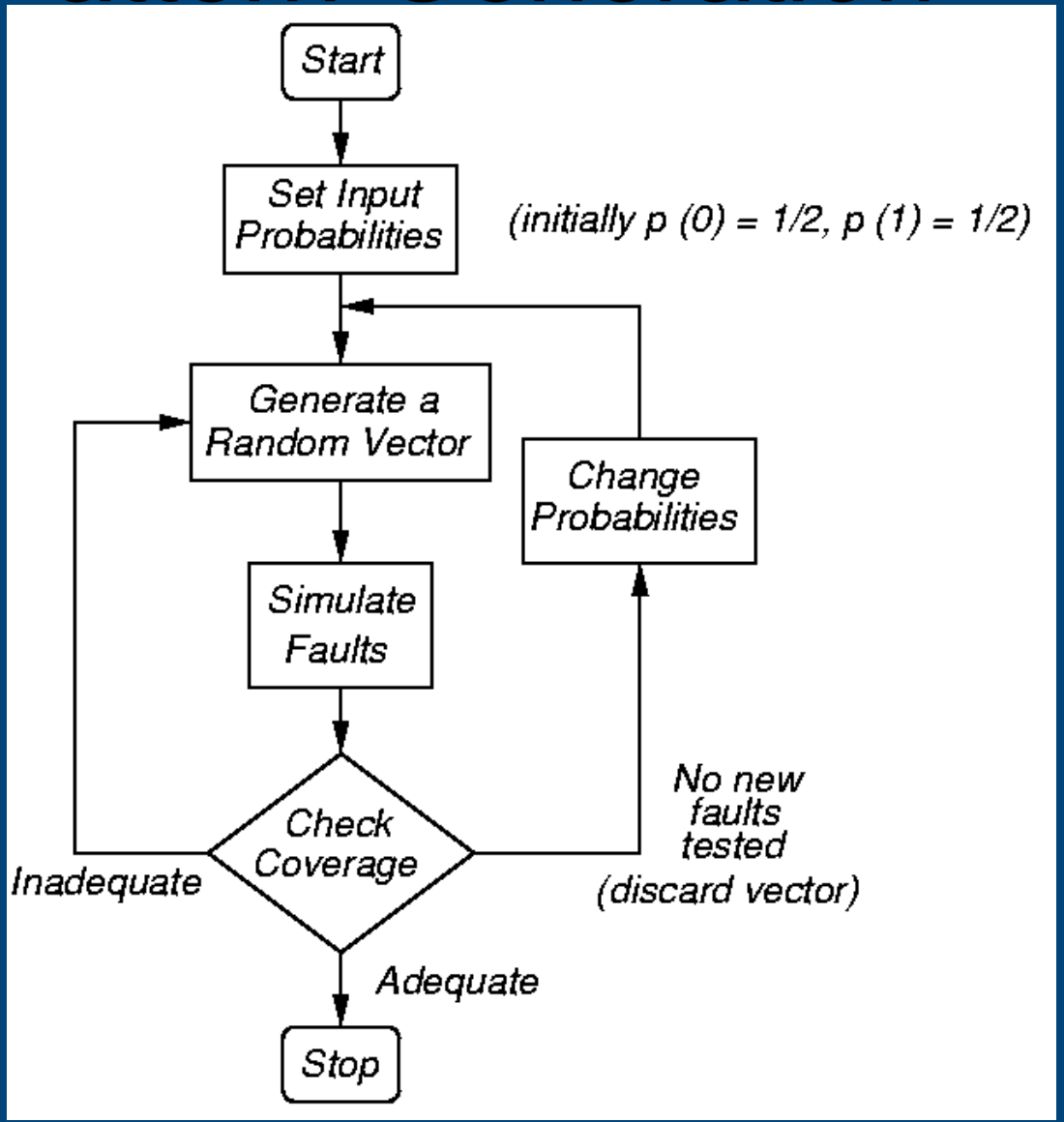
- Exhaustive
  - cheap generation, high FC, expensive application
- Fault-Oriented (deterministic)
  - expensive generation, possibly high FC, cheaper application
  - reduction of generation costs
- Random (pseudo-random)
  - cheap generation, low FC, + - expensive application

Exhaustive Algorithm

- For *n*-input circuit, generate all 2^n input patterns
- Infeasible, unless circuit is partitioned into cones of logic, with ≤ 15 inputs
  - Perform exhaustive ATPG for each cone
  - Misses faults that require specific activation patterns for multiple cones to be tested

Random-Pattern Generation

- Flow chart for method
- Use to get tests for 60-80% of faults, then switch to D-algorithm or other ATPG for rest

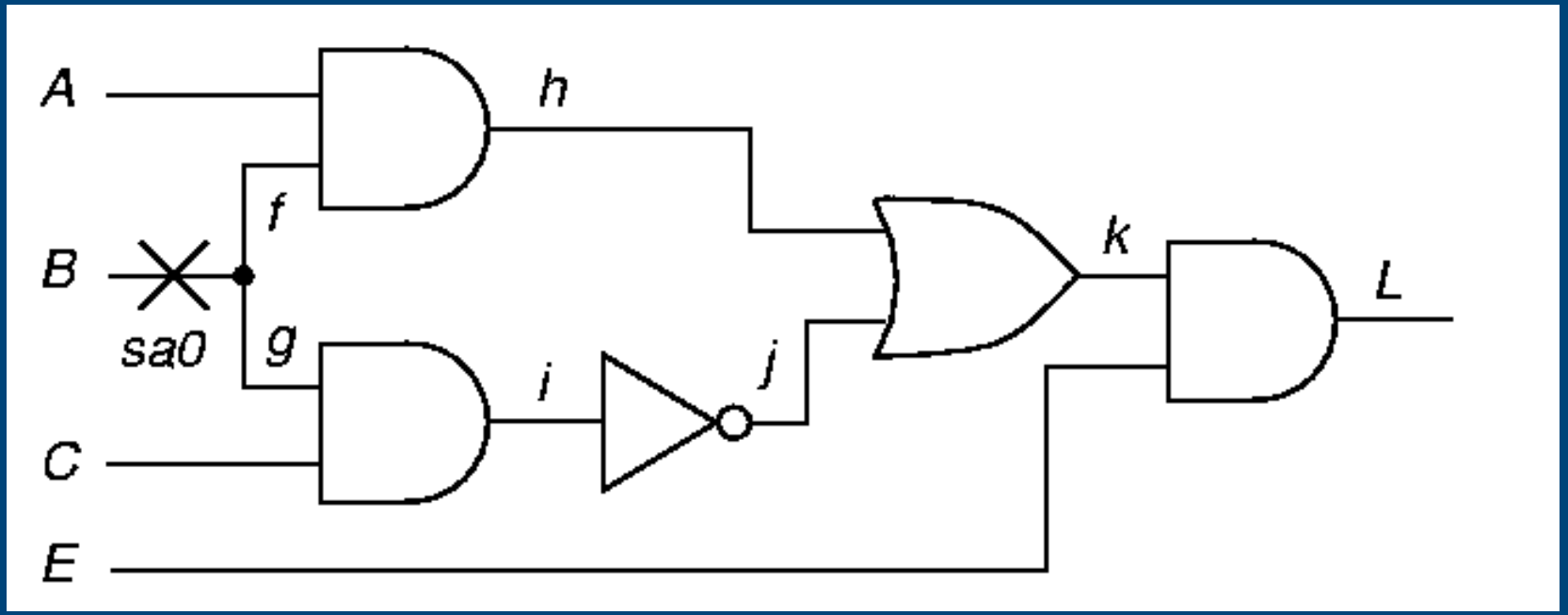


Path Sensitization Method

1 Fault Sensitization (activation)

2 Fault Propagation

3 Line Justification



Path Sensitization Method

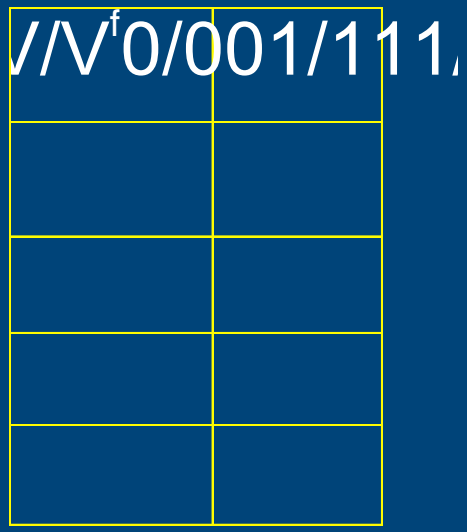
- Fault *l* *s-a-v*
- Activation
  - set *l* to $\overline{v}$
- Propagation
  - find a path from *l* to a primary output that keeps faulty value
- Justification
  - set the primary inputs to activate the fault

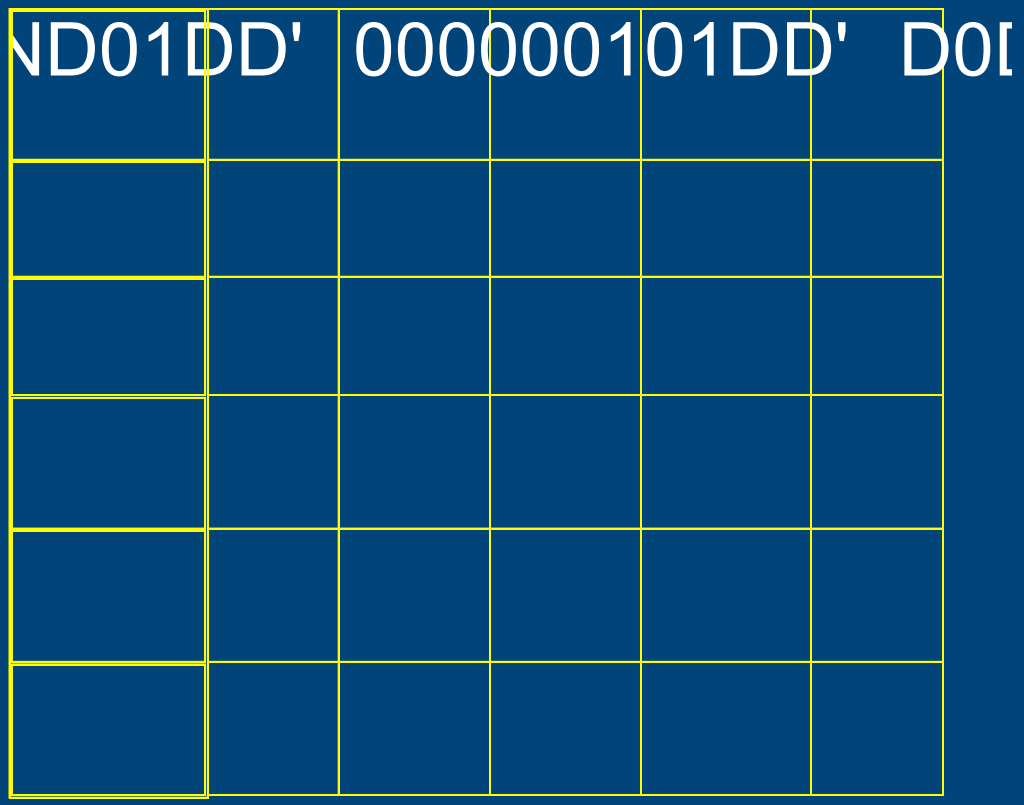
Composite Logic Values

- consider line value for original AND faulty circuit
- v/v_f = original/faulty
- Symbols D and $\overline{D}$ (Roth, 1966)
- D = 1/0
- $\overline{D}$ = 0/1
- 0 = 0/0
- 1 = 1/1

Operations on Composite Values

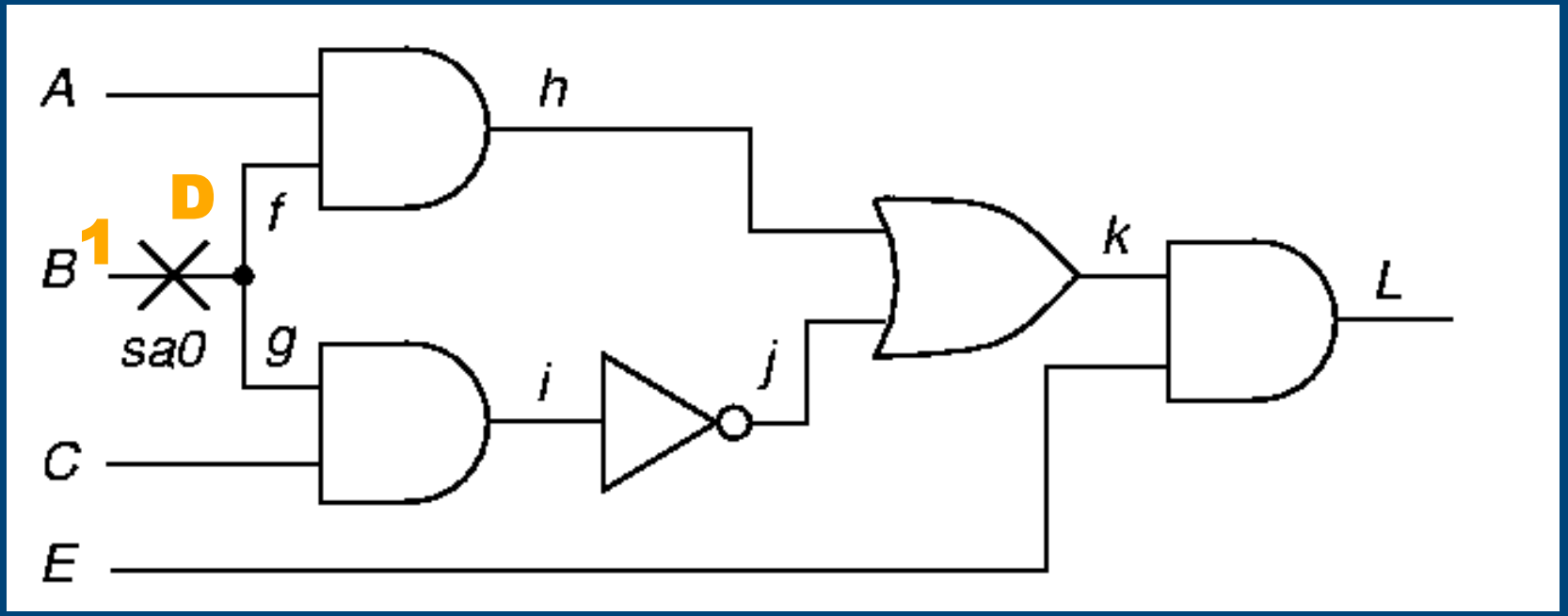
- $\overline{D} + 0 = 0/1 + 0/0 = 0/1 = \overline{D}$





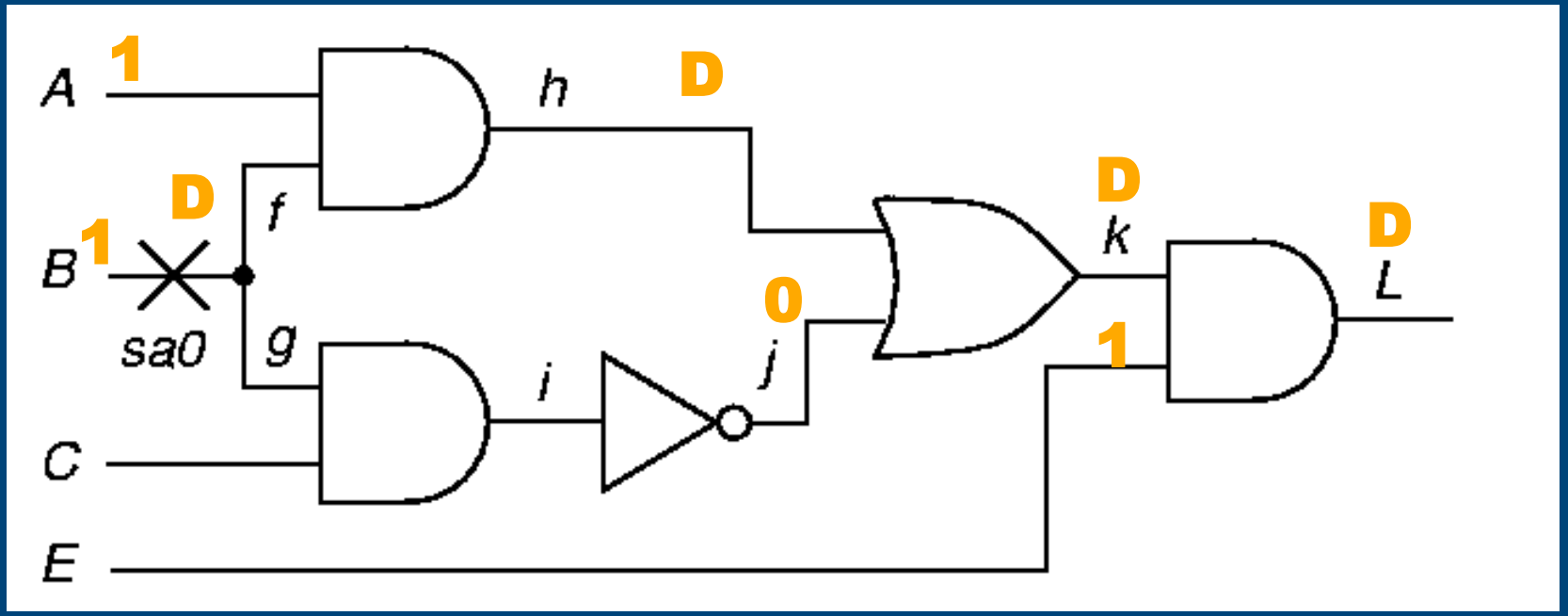
Path Sensitization Method

- Propagation



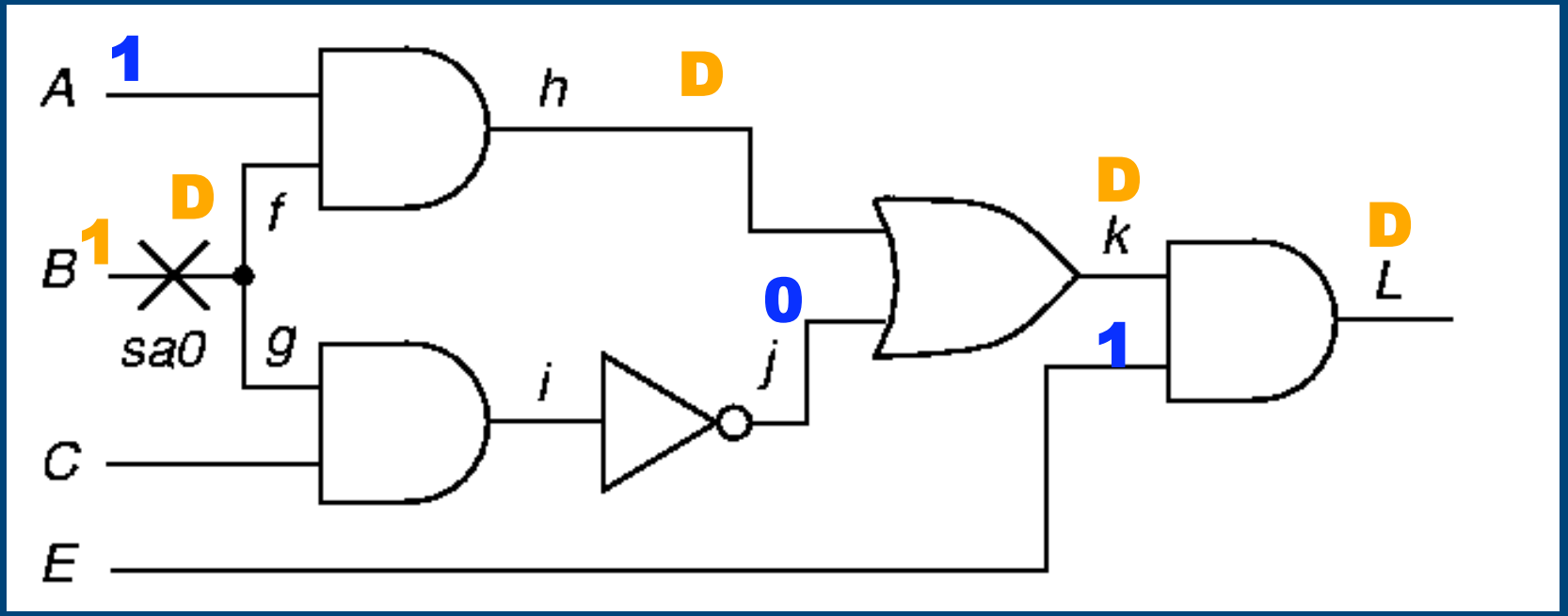
Path Sensitization Method

- Propagation: try path *f* – *h* – *k* – *L*



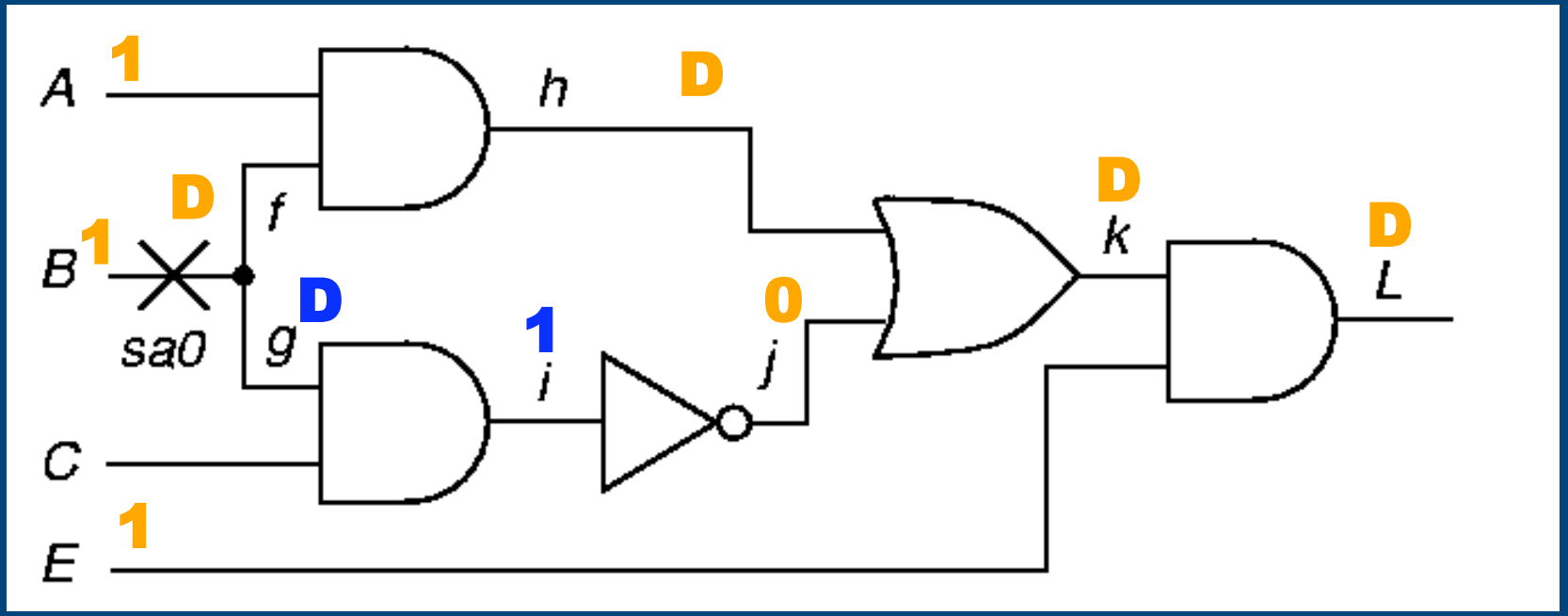
Path Sensitization Method

- Propagation: try path *f* – *h* – *k* – *L*



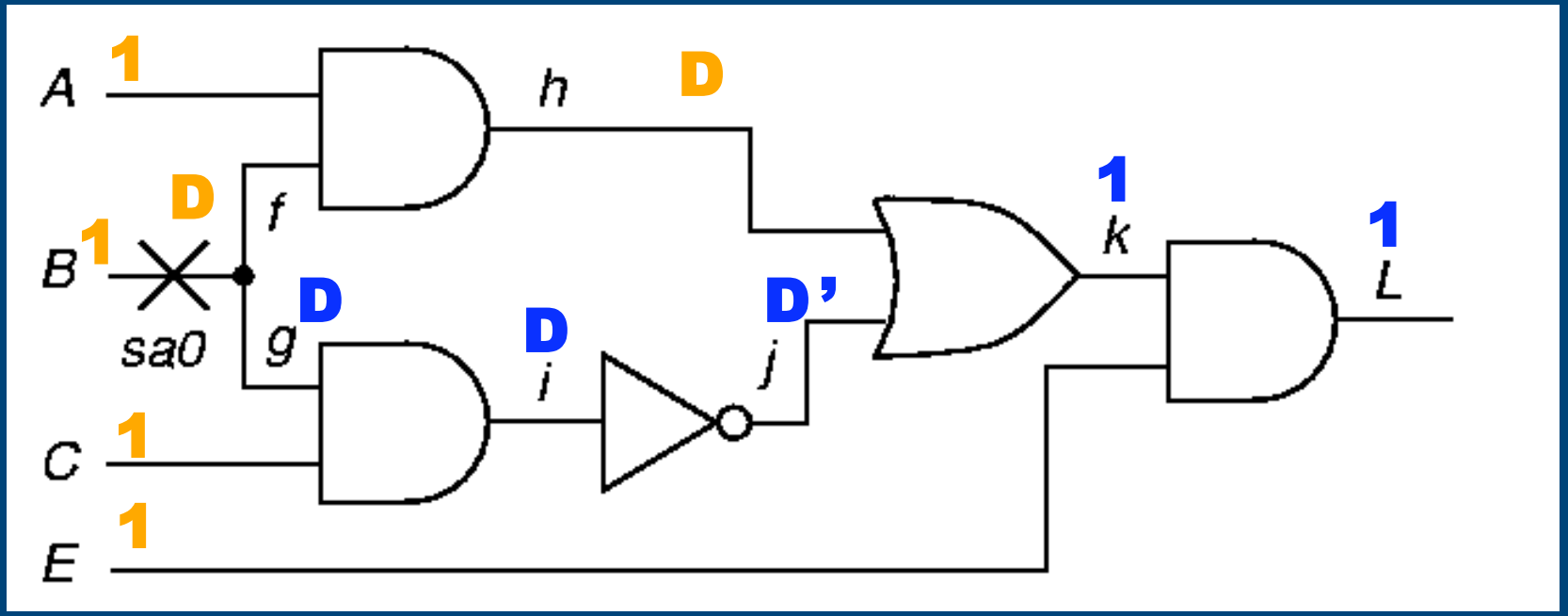
Path Sensitization Method

- Justification: Try path *f* – *h* – *k* – *L* blocked at *j*, since there is no way to justify the 1 on *i*



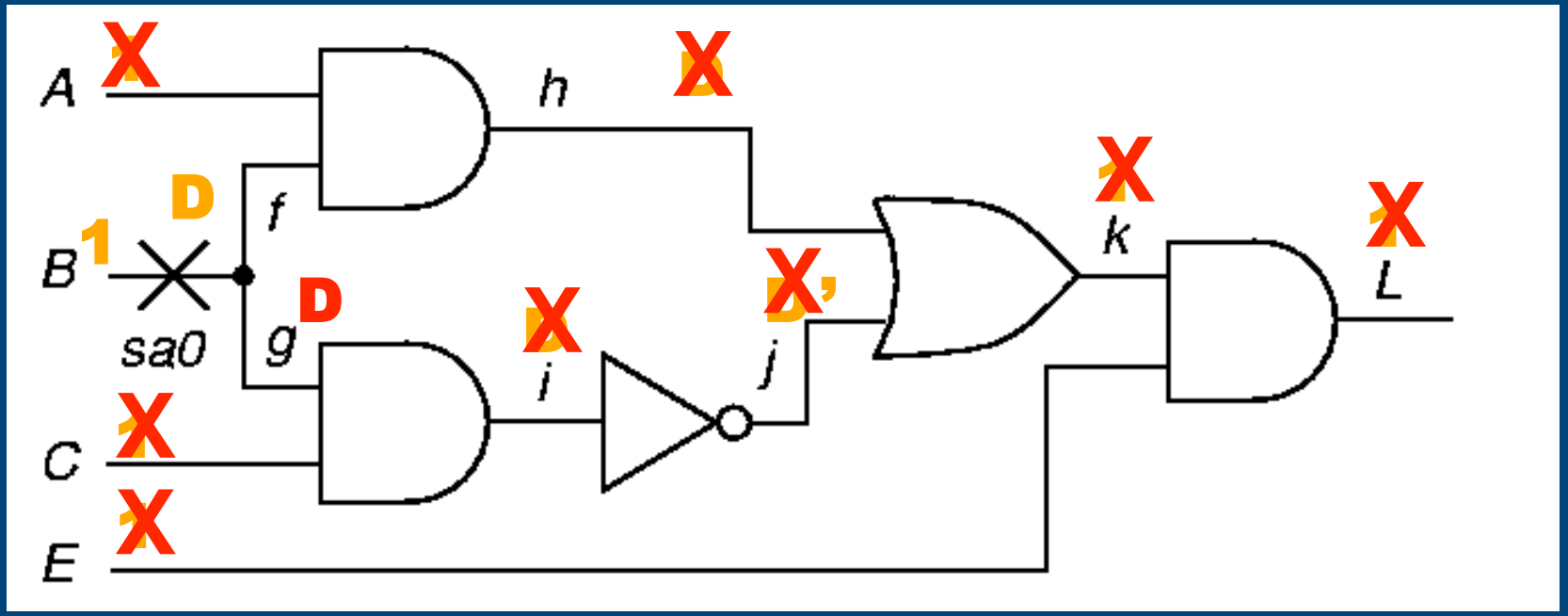
Path Sensitization Method

- Justification: Try path *f* – *h* – *k* – *L* blocked at *j*, since there is no way to justify the 1 on *i*



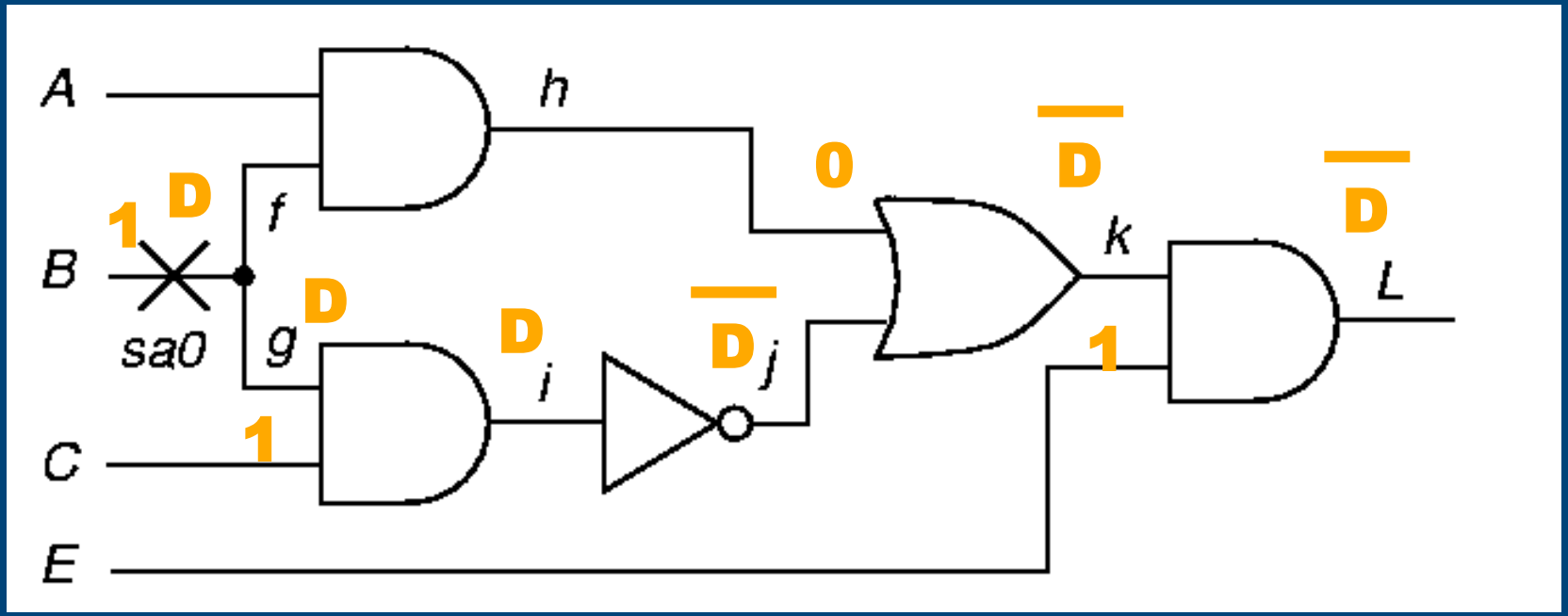
Path Sensitization Method

- Backtracking!



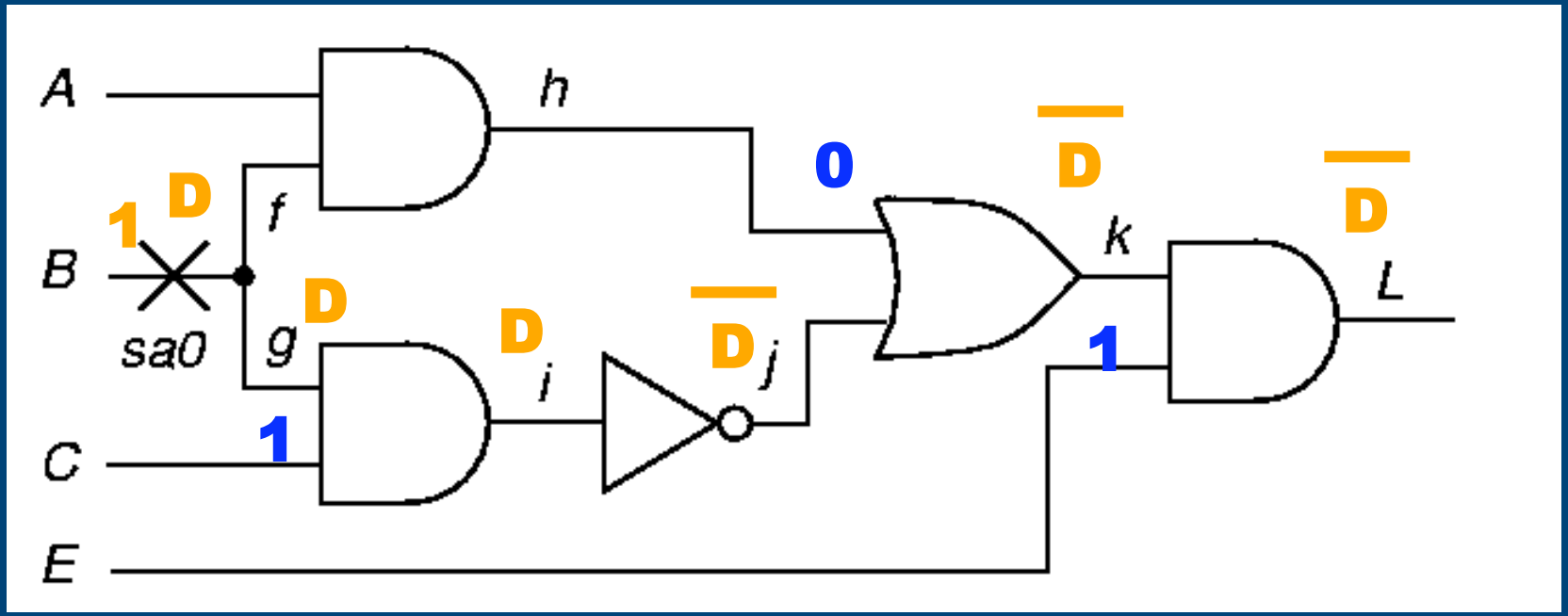
Path Sensitization Method

- Try other propagation: path $g - i - j - k - L$



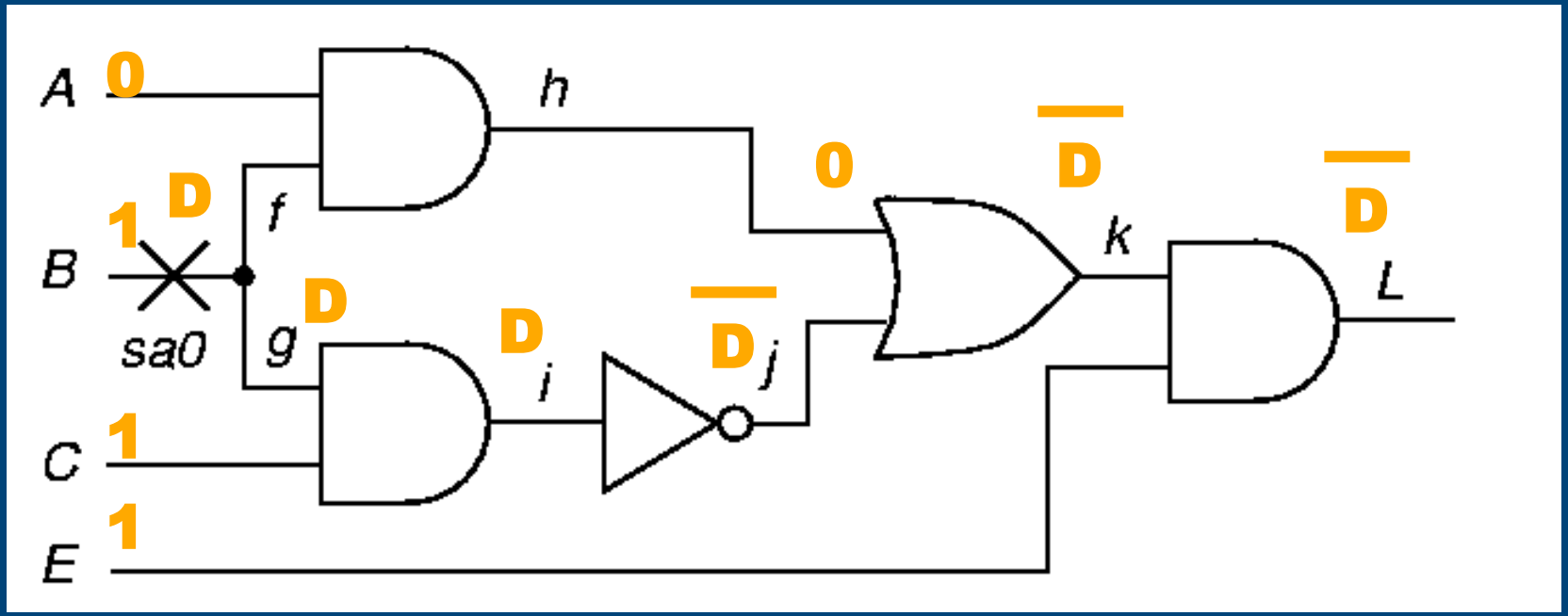
Path Sensitization Method

- Try other propagation: path $g - i - j - k - L$



Path Sensitization Method

- Try other propagation: path $g - i - j - k - L$



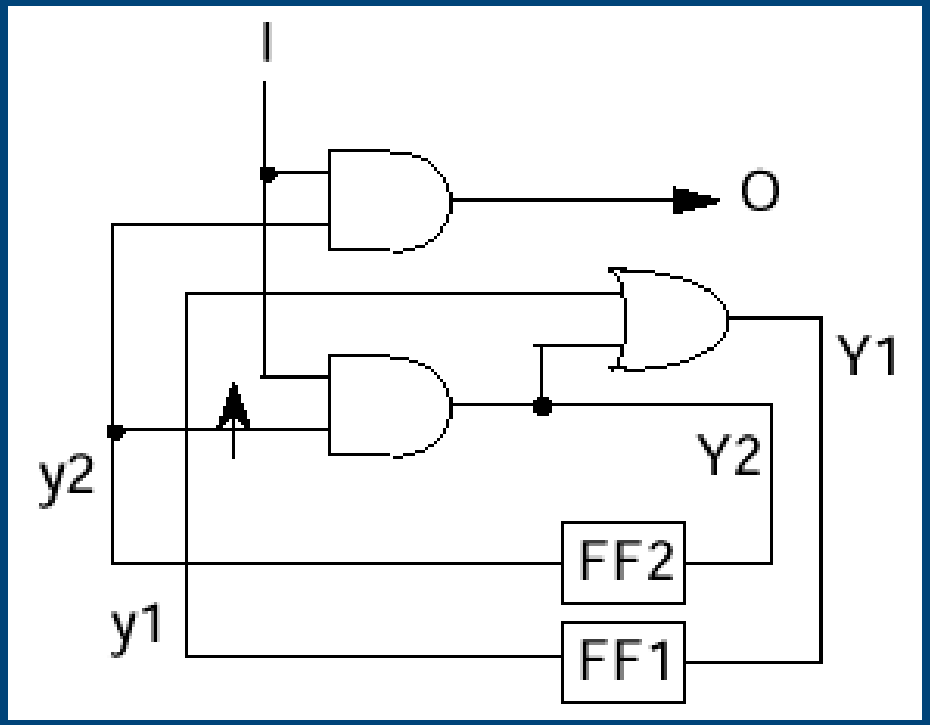
Major Combinational Automatic Test-Pattern Generation Algorithms

- D-Algorithm (Roth) -- 1966
- PODEM (Goel) -- 1981
- FAN (Fujiwara and Shimono) --1983

Sequential Circuit ATPG
Time-Frame Expansion

- Problem of sequential circuit ATPG
- Time-frame expansion

Example of Sequential Circuit



Sequential Circuits

- A sequential circuit has memory in addition to combinational logic.
- Test for a fault in a sequential circuit is a sequence of vectors, which
  - Initializes the circuit to a known state
  - Activates the fault, and
  - Propagates the fault effect to a primary output
- Methods of sequential circuit ATPG
  - Time-frame expansion methods
  - Simulation-based methods

Extended D-Algorithm

1. Pick up a target fault f.

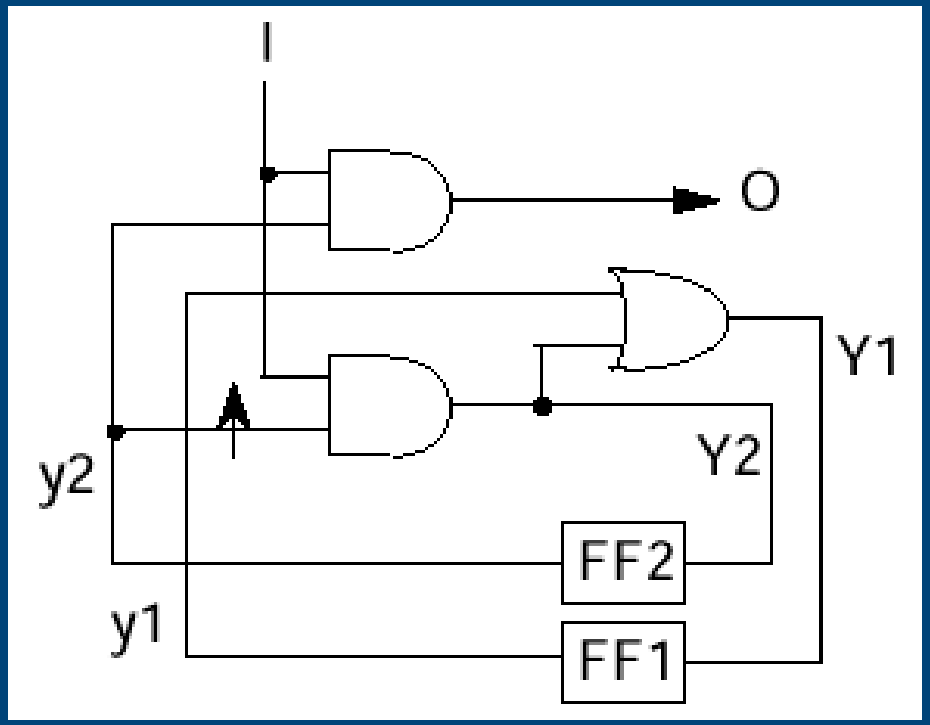
2. Create a copy of a combinational logic, set it time-frame 0.

3. Generate a test for f using D-algorithm for time-frame 0.

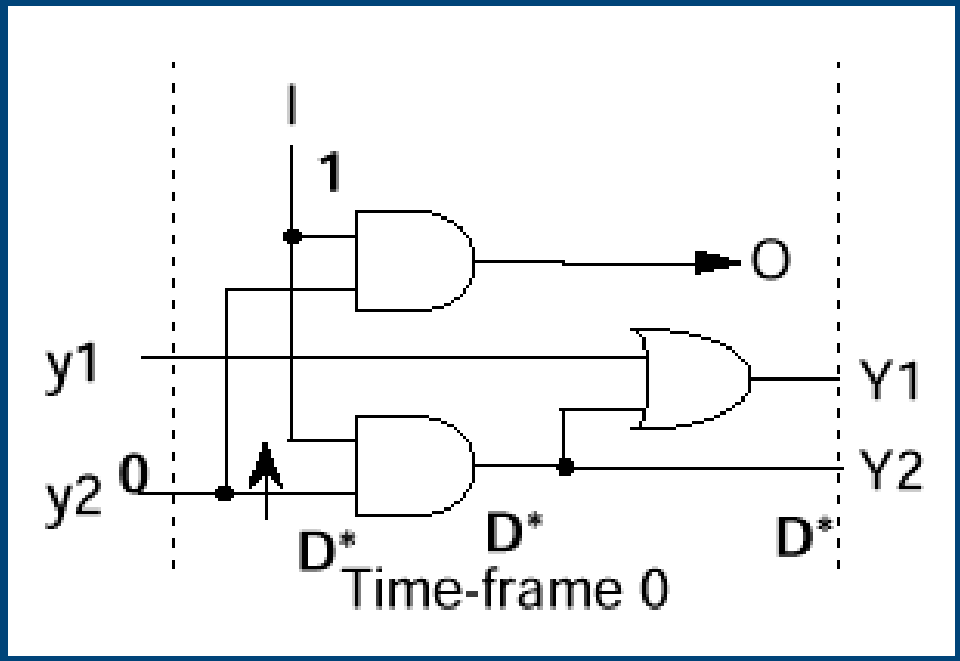
4. When the fault effect is propagate to the DFFs, continue fault propagation in the next time-frame.

5. When there are values required in the DFFs, continue the justification in the previous time-frame.

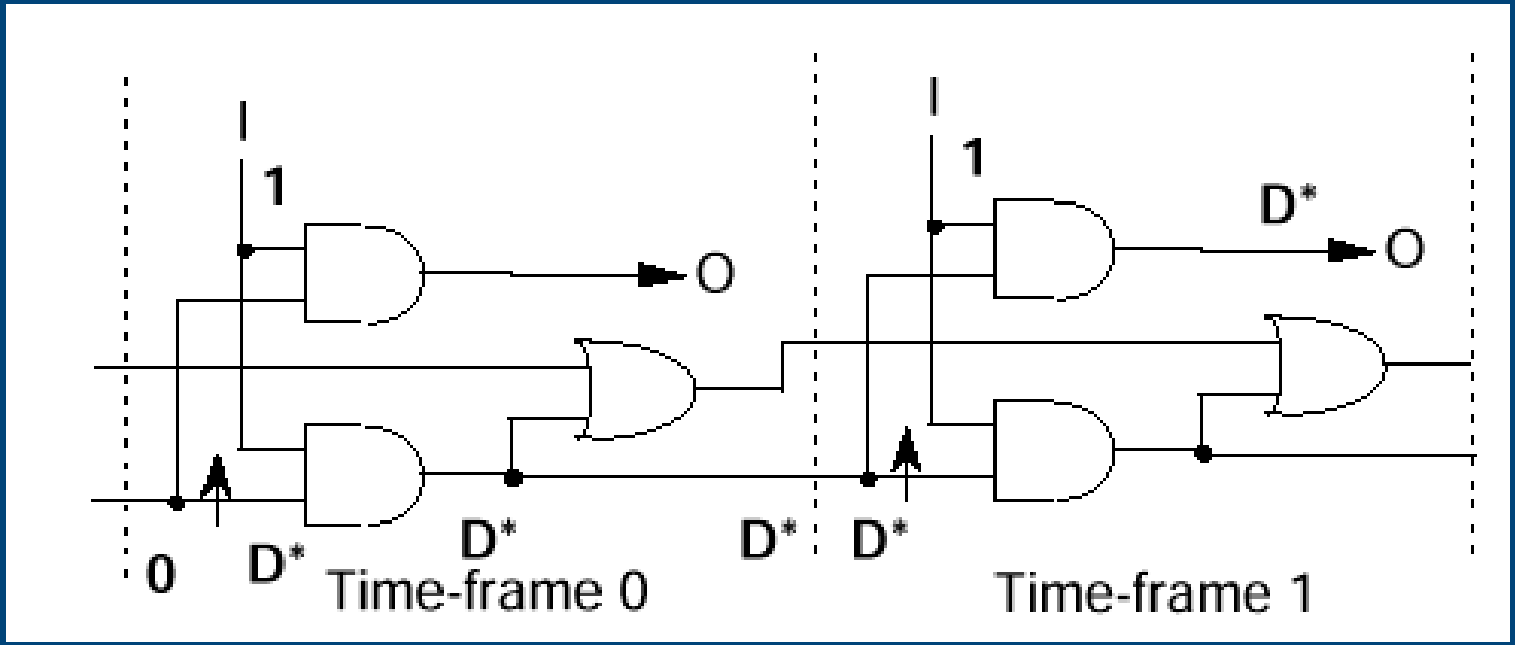
Example for Extended D- Algorithm



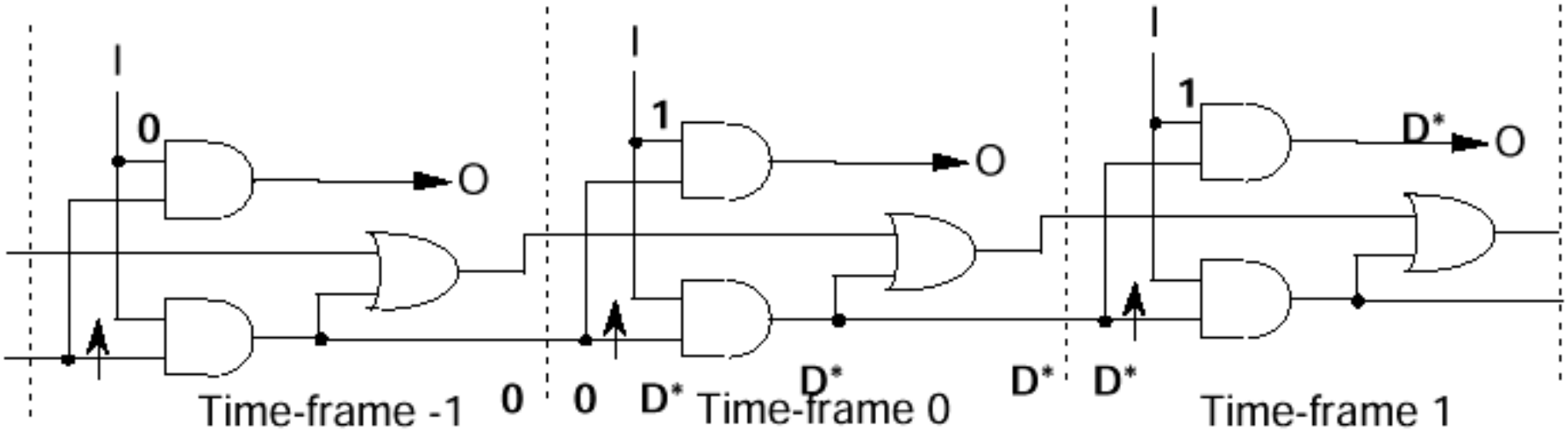
Example: Step 1



Example: Step 2



Example: Step 3



Summary

- Hierarchical ATPG -- 9 Times speedup (Min)
  - Handles adders, comparators, MUXes
- Advances over D-algorithm

- Results of 40 years research – mature – methods:
  - Path sensitization
  - Simulation-based
  - Boolean satisfiability and neural networks
  - Genetic algorithms