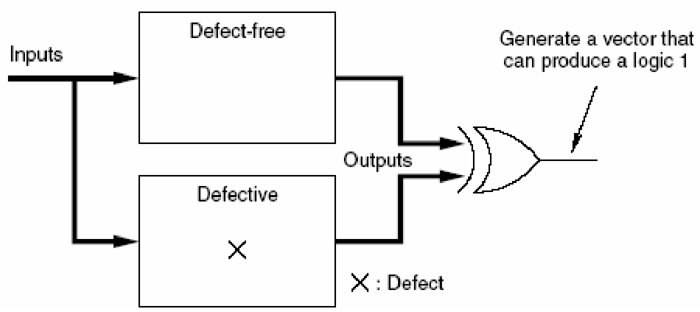
CPE 628
Chapter 4 – Test Generation

Dr. Rhonda Kay Gaede

UAH

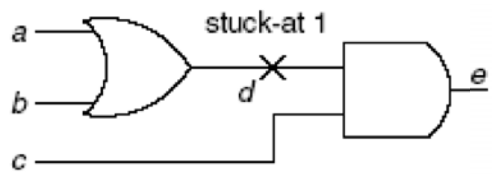
4.1 Introduction – Conceptual View

Generate an input vector that can ___________ the _____-___ circuit from the _________ one



4.1 Introduction – Simple Illustration

- ❑Consider the fault d/1 in the defective circuit
- ❑To detect
  - ❑______ Fault, set d = __, means that __ = __ = 0
  - ❑__________ Fault Effect, set c = __
- ❑Vector: abc=_____ (output = __/__)



4.1 Introduction – Typical ATPG System

- ❑Given a circuit and a _____ ______
  - ▪Repeat
    - ▪Generate a ____ for each _________ fault
    - ▪_____ all other faults detected by the test using a ____ _________
  - ▪Until ___ _______ have been considered
- ❑Note 1: a fault may be ___________, so no test can be generated
- ❑Note 2: an ATPG may abort on a fault if the _______________ needed exceed a preset limit

4.2 Random Test Generation –

- ❑__________ form of test generation
  - ▪*N* tests are randomly generated
- ❑Level of confidence on random test set *T*
  - ▪The probability that *T* can detect ___ stuck-at faults in the given circuit
  - ▪______ of a random test set highly depends on the underlying circuit
  - ▪Some circuits have many ______-________ faults

4.2 Random Test Generation – Adding Weights

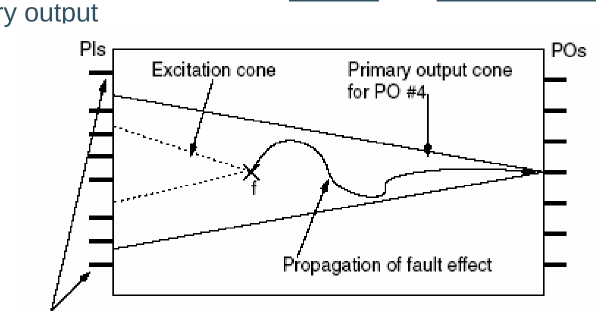
- ❑____ input probabilities to _____ random resistant faults
- ❑Consider an 8-input AND gate
  - ▪Without biasing input probabilities, the probability of generating a logic 1 at the gate output = $(0.5)^8 = 0.004$
  - ▪If we bias the inputs to 0.75, then the probability of generating a logic 1 at the gate output = $(0.75)^8 = 0.100$
- ❑Obtaining an _________ set of input probabilities a difficult task
- ❑Goal: increase the signal probabilities of ____-__-____ regions

4.2 Random Test Generation – Probability of Fault Detection

- Given a circuit with *n* inputs
- Let T_f be the ___ __ _______ that can detect fault *f*
- Then $d_f = \frac{T_f}{2^n}$ is the probability that *f* can be ________ by a ______ vector
- Let $e_f = 1 - d_f$ be the probability that a random vector ______ detect *f*
- Then, $e_f^N = (1 - d_f)^N$ is the probability that __ random vectors do ___ detect *f*
- Thus, the probability that __ ____ one out of *N* random vectors can detect *f* is $1 - (1 - d_f)^N$

4.2 Random Test Generation – Minimum Detection Probability

- The ________ detection probability of any detectable fault actually does not depend on *n*, the num of ____s
- Instead, it depends on its largest ________-______ ____
- Any detectable fault must be ______ and _____________ to a primary output



Inputs outside of PO cone are not needed for detection of fault f

4.2 Random Test Generation – Exhaustive Test Generation

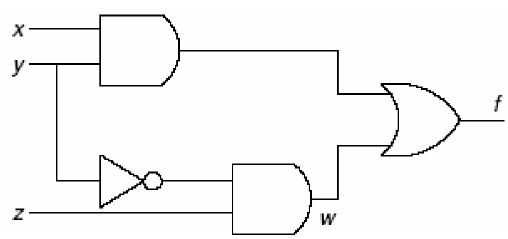
❑Exhaustive Testing

- Apply ___ patterns to an __-input combinational *circuit under test* (CUT)
- ______________ all detectable faults in the combinational circuits are detected
- Test time maybe be _____________ ____ if the number of inputs is ______
- Feasible only for _____ circuits

❑Pseudo-exhaustive Testing

- ________ circuit into respective cones
- Apply _____________ testing only to each cone

4.3 Theoretical Background: Boolean Difference – Definition

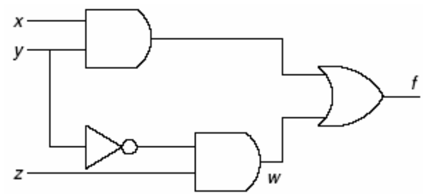


❑For a function $f(x_1, x_2, ..., x_n)$, the Boolean Difference is defined as $\frac{df}{dx_i} = f(x_1, x_2, ..., x_i, ...x_n) \oplus f(x_1, x_2, ..., \overline{x_i}, ...x_n)$

❑The Boolean Difference defines how a fault effect on x_i can be propagated to an observable output

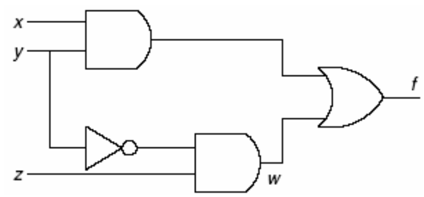
4.3 Theoretical Background: Boolean Difference – Test Generation

- To detect x_i s-a0, $x_i \bullet \frac{df}{dx_i} = 1$
- To detect x_i s-a-1, $\overline{x_i} \bullet \frac{df}{dx_i} = 1$
- Consider y s-a-0



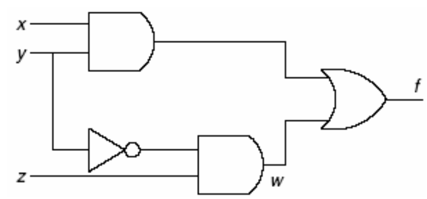
4.3 Theoretical Background: Boolean Difference – Test Generation(Another Fault)

- Consider w s-a-0



4.3 Theoretical Background: Boolean Difference – Untestable Faults

- Consider z s-a-0



4.4 Designing a Stuck-at ATPG for Combinational Circuits

- In general, we don't need an ______ ___ of vectors that can detect the target fault
- Instead, we just want to compute ___ vector quickly
- Rather than using Boolean Difference that can obtain all vectors
  - Simply use a ______-and-______ search to find one vector quickly
- Deterministic ATPG has two main goals
  - ______ the target fault
  - _________ the corresponding fault effect to an output

4.4 Designing a Stuck-at ATPG for Combinational Circuits – D-algebra

- For ATPG, we want to express ______ and ____-___ behavior at the same time, use a ___
- 5-Value Algebra: 0(__), 1(__), X(__), D(__), D'(__).

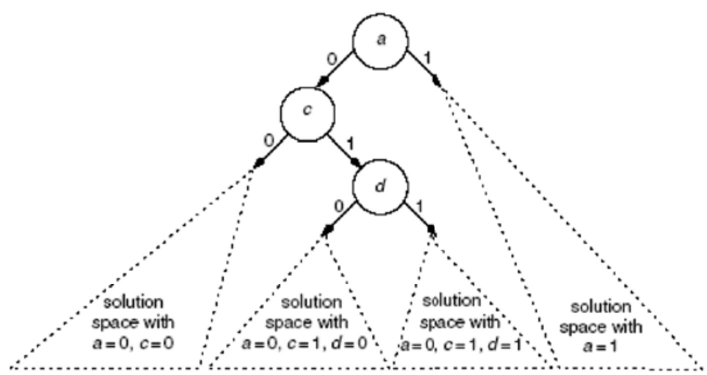
AND	0	1	D	$\overline{D}$	X
0	0	0	0	0	0
1	0	1	D	$\overline{D}$	X
D	0	D	D	0	X
$\overline{D}$	0	$\overline{D}$	0	$\overline{D}$	X
X	0	X	X	X	X

OR	0	1	D	$\overline{D}$	X
0	0	1	D	$\overline{D}$	X
1	1	1	1	1	1
D	D	1	D	1	X
$\overline{D}$	$\overline{D}$	1	1	$\overline{D}$	X
X	X	1	X	X	X

NOT	
0	1
1	0
D	$\overline{D}$
$\overline{D}$	D
X	X

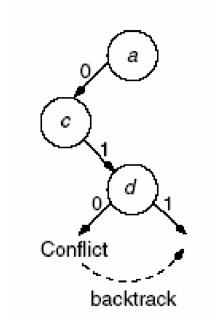
4.4 Designing a Stuck-at ATPG for Combinational Circuits – Decision Tree

The ATPG systematically and implicitly searches the entire search space



4.4 Designing a Stuck-at ATPG for Combinational Circuits – Backtracking

Whenever a ________ is found, we must go back to the _____ _________ made and make a different one, this process is called ____________.



4.4 Designing a Stuck-at ATPG for Combinational Circuits- Algorithm

Algorithm 2 Basic Fanout Free ATPG (C, g/v)

1: initialize circuit by setting all values to X;
2: JustifyFanoutFree($C, g, \overline{v}$); /* excite the fault by justifying line g to $\overline{v}$ */
3: PropagateFanoutFree(C, g); /* propagate fault-effect from g to a PO */

UAH | Chapter 4 | CPE 628

4.4 Designing a Stuck-at ATPG for Combinational Circuits- Justify

Algorithm 3 JustifyFanoutFree(C, g, v)

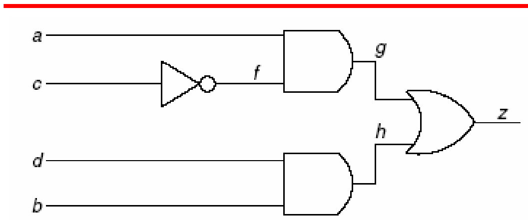
```
1: g = v;
2: if gate type of g == primary input then
3:    return;
4: else if gate type of g == AND gate then
5:    if v == 1 then
6:       for all inputs h of g do
7:          JustifyFanoutFree(C, h, 1);
8:       end for
9:    else {v == 0}
10:      h = pick one input of g whose value == X;
11:      JustifyFanoutFree(C, h, 0);
12:   end if
13: else if gate type of g == OR gate then
14:   ...
15: end if
```

Justify works from a ___ ______ ______ towards the primary inputs.

____________ ___________ might be used in step 10

UAH | Chapter 4 | CPE 628

4.4 Designing a Stuck-at ATPG for Combinational Circuits- Justify Example



Consider g/0

Justify called 4 times

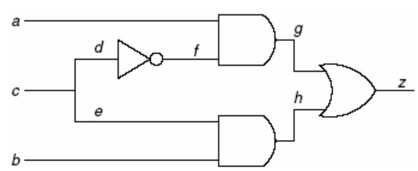
- **JustifyFanoutFree(C, __, __)**
- **JustifyFanoutFree(C, __, __)**
- **JustifyFanoutFree(C, __, __)**
- **JustifyFanoutFree(C, __, __)**

Consider z/1

- **JFF(C, __, __)**
- **JFF(C, __, __)**
- **JFF(C, __, __)**

4.4 Designing a Stuck-at ATPG for Combinational Circuits- Justify Example #2

Consider g/0



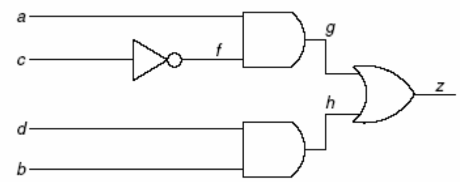
Then d/0

4.4 Designing a Stuck-at ATPG for Combinational Circuits- Propagate

Algorithm 4 PropagateFanoutFree(*C*, *g*)

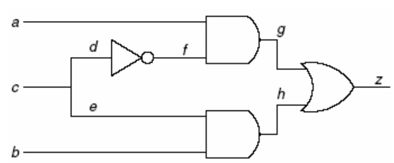
```
if g has exactly one fanout then
    h = fanout gate of g;
    if none of the inputs of h has the value of X then
        backtrack;
    end if
else {g has more than one fanout}
    h = pick one fanout gate of g that is unjustified;
end if
if gate type of h == AND gate then
    for all inputs, j, of h, such that j ≠ g do
        if the value on j == X then
            JustifyFanoutFree(C, j, 1);
        end if
    end for
else if gate type of h == OR gate then
    for all inputs, j, of h, such that j ≠ g do
        if the value on j == X then
            JustifyFanoutFree(C, j, 0);
        end if
    end for
else if gate type of h == . . . gate then
    . . .
end if
PropagateFanoutFree(C, h);
```

4.4 Designing a Stuck-at ATPG for Combinational Circuits - Propagate Example



Back to g/0, after justify sets g to 1, then

4.4 Designing a Stuck-at ATPG for Combi-national Circuits - Propagate Example #2

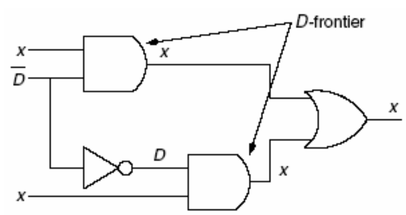


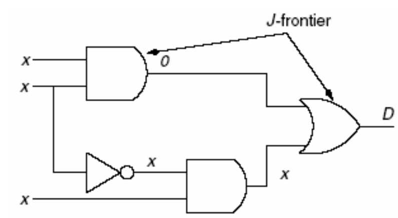
Consider g/0

Consider g/1

4.4 Designing a Stuck-at ATPG for Combinational Circuits - D Algorithm

- The D algorithm tries to ___________ a D or D’ to a ________ _______ by making assignments on _________ signals and primary inputs.
  - _________, gates with a D or D’ on the input but not on the output
- Once D or D’ makes it to a ________ ______, the algorithm tries to ________ the values used for _______________
  - ________, gates with a value set on the output but not justified by its inputs





4.4 Designing a Stuck-at ATPG for Combinational Circuits - D Algorithm

Algorithm 5 *D*-Algorithm(*C*, *f*)

```
initialize all gates to don't-cares;
set a fault-effect (D or D̄) on line with fault f and insert it to the D-frontier;
J-frontier = φ;
result = D-Alg-Recursion(C);
if result == success then
    print out values at the primary inputs;
else
    print fault f is untestable;
end if
```

4.4 Designing a Stuck-at ATPG for Combinational Circuits - D-Alg-Recursion

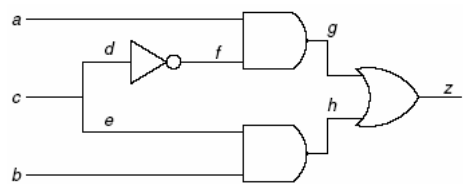
```
if there is a conflict or D-frontier is ϕ then
      return (failure);
end if
/* first propagate the fault-effect to a PO */
if no fault-effect has reached a PO then
      while not all gates in D-frontier have been tried do
            G = an untried gate in D-frontier
            set all unassigned inputs of g to non-controlling and
                  add to J-frontier
            result = D-Alg-Recursion(C);
            if result == success then
                  return (success);
            end if
      end while
      return (failure);
end if    /*fault effect has reached at least one PO*/
```


4.4 Designing a Stuck-at ATPG for Combinational Circuits - D-Alg-Recursion

```
if j-frontier is ϕ then
      return (success);
end if
g = a gate in J-frontier
while g has not been justified do
      j = an unassigned input of g;
      set j = 1 and insert j = 1 to J-frontier
      result = D-Alg-Recursion(C);
      if result == success then
            return(success);
      else try the other assignment
            set j = 0;
      end if
end while
return(failure);
```

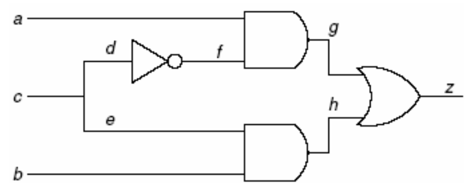
4.4 Designing a Stuck-at ATPG for Combi-national Circuits - D-Algorithm Example #1

Target fault: f/0



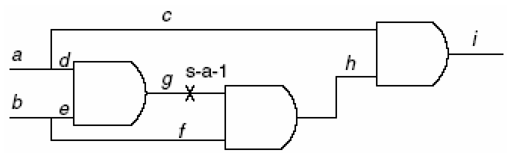
4.4 Designing a Stuck-at ATPG for Combi-national Circuits - D-Algorithm Example #2

Target fault: f/1



4.4 Designing a Stuck-at ATPG for Combinational Circuits - D-Algorithm Example #3

Target fault: g/1



4.4 Designing a Stuck-at ATPG for Combinational Circuits - PODEM

- In the D algorithm, the ________ space consists of ____ the lines in the circuit.
- PODEM makes _________ only at the _______ _______, eliminating any unjustified values
  - Backtracks when D-frontier is ______
- Picks an objective (___________ or ___________) and traces it back to a primary input _______ making assignment

4.4 Designing a Stuck-at ATPG for Combinational Circuits - PODEM

Algorithm 9 getObjective(C)

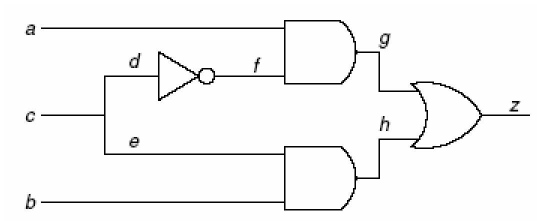
if fault is not excited **then**
    return $(g, \overline{v})$;
end if
$d =$ a gate in D-frontier;
$g =$ an input of d whose value is x;
$v =$ non-controlling value of d;
return (g, v);

4.4 Designing a Stuck-at ATPG for Combinational Circuits - PODEM

Algorithm 10 backtrace(C)

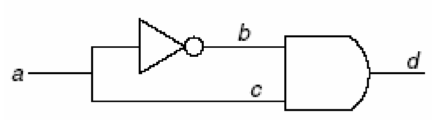
$i = g$;
num_inversion $= 0$;
while $i \neq$ primary input **do**
    $i =$ an input of i whose value is x;
    if i is an inverted gate type **then**
        num_inversion++;
    end if
end while
if num_inversion == odd **then**
    $v = \overline{v}$;
end if
return(i, v);

4.4 Designing a Stuck-at ATPG for Combinational Circuits – PODEM Example



- 1st Objective:
- Backtrace from the objective:
- Simulate(c=0): D-Frontier =
- 2nd Objective:
- Backtrace from the objective:
- Simulate(a=0):

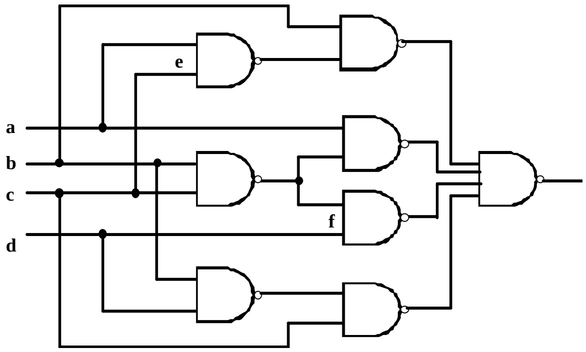
4.4 Designing a Stuck-at ATPG for Combinational Circuits – PODEM Example



Target fault: b/0

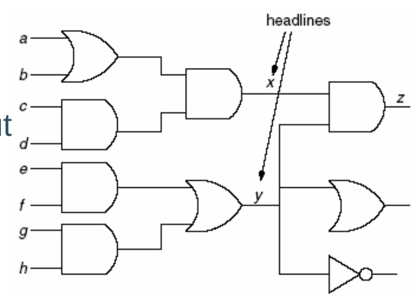
- 1st Objective:
- Backtrace from objective:
- Simulate(a=0):
- Must backtrack
- Change decision
- Simulate(a=1):
- Backtrack

4.4 Designing a Stuck-at ATPG for Combinational Circuits – PODEM Example



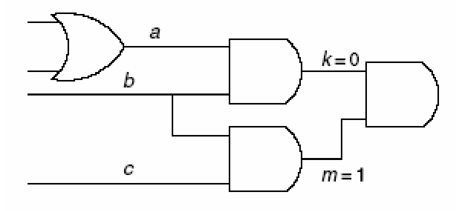
4.4 Designing a Stuck-at ATPG for Combinational Circuits – FAN

- ______ PODEM for an improved ATPG
  - ________ the number of decision points ________
- Concept of __________
  - A __________ is the output of a ______ _____ region, backtrace can ______ _____ from a ________ to a primary input
- _________ Objectives to reduce later ________



UAH Chapter 4 CPE 628

4.4 Designing a Stuck-at ATPG for Combinational Circuits – FAN



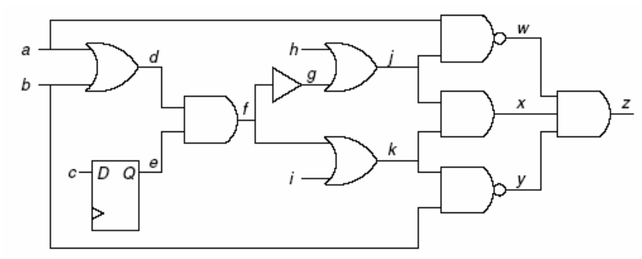
- Objectives:
- Backtrace from k=0 may favor ____, but __________ would _________ the second objective m=1!
- Choose _____ instead
- Makes backtrace more ___________ to avoid future conflicts

UAH Chapter 4 CPE 628

4.4 Designing a Stuck-at ATPG – Static Logic Implications

- ❑________ logic implications
- ❑__________ logic implications
- ❑___________ __________ implications

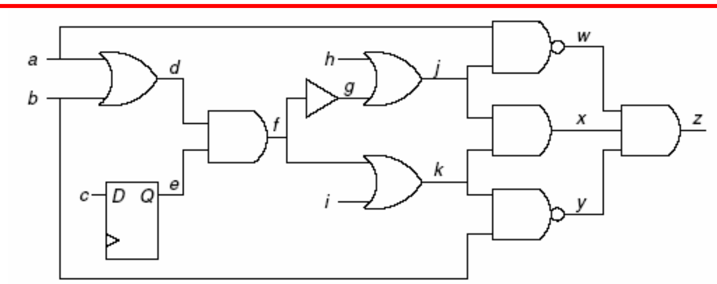
4.4 Designing a Stuck-at ATPG – Static Logic Implications (Direct)



- Direct implications for f=1:

- Direct implications for j=0:

4.4 Designing a Stuck-at ATPG – Static Logic Implications (Indirect)

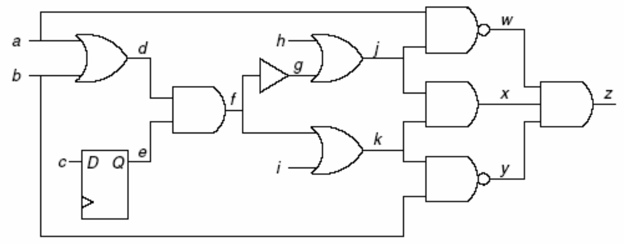


- Direct implications for f=1:

- Indirect Implications for f=1 obtained by simulating the direct implications of f=1:

- This process is repeated for every node in the circuit

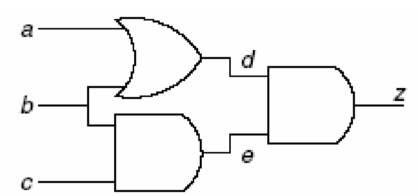
4.4 Designing a Stuck-at ATPG – Static Logic Implications (Extended Backwards)



- In order to justify ____, need either ____ or ____
  - Simulate(a=1, impl(f=1)) = Sa
  - Simulate(b=1, impl(f=1)) = Sb
- ___________ of Sa and Sb is the the set of extended backward mplications for f=1
- This process is repeated for every _________ gate, as well as for every node in the circuit

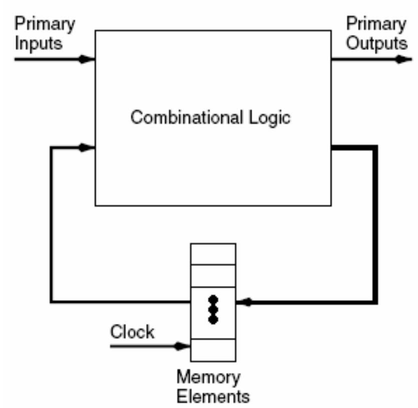
4.4 Designing a Stuck-at ATPG – Dynamic Logic Implications

- Similar to ______ Logic Implications, but some signals _______ ____ assigned values
- Suppose ____ has already been assigned
  - Then to obtain z=0, _____________
  - d=0 requires _________, e=0 requires ________
  - The intersection of _________ and ______ is _____



UAH Chapter 4 CPE 628

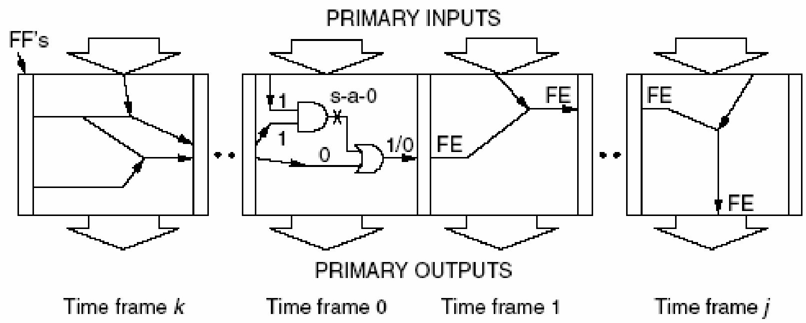
4.5 Sequential ATPG – Huffman Model



Electrical and Computer Engineering Page 45

UAH Chapter 4 CPE 628

4.5 Sequential ATPG – Iterative Logic Array Expansion



To detect a fault, a sequence of vectors may be needed

Electrical and Computer Engineering Page 46

UAH Chapter 4 CPE 628

4.5 Sequential ATPG – Basic Framework

- Based on _______________ ATPG
- Targets one __________ at a time
- Excite the target fault in time-frame __ and propagate it to a __, possibly through _________ time-frames
- _______ the state needed at time-frame __, via possibly several time-frames
- Sequential ATPG very complex, as backtracks can involve reversing decisions at different time-frames

Electrical and Computer Engineering

Page 47

UAH Chapter 4 CPE 628

4.6 Untestable Fault Identification

- Untestable faults are:
  - Those that could not be ________, or
  - Those that could not be __________, or
  - Those that could not be ________________ _______ and _________
- ATPG can spend a lot of time trying to generate a test for an untestable fault

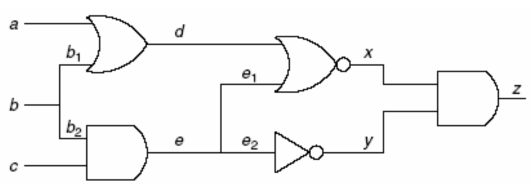
Electrical and Computer Engineering

Page 48

4.6 Untestable Fault Identification – FIRE Method

- Based on _______ analysis
- S0 = set of faults that are untestable when signal ______
- S1 = set of faults that are untestable when signal ______
- _____________ of S0 and S1 are definitely untestable
  - They require s=1 and s=0 ____________ to be detectable!
- Propagate ________________
- Propagate ________________

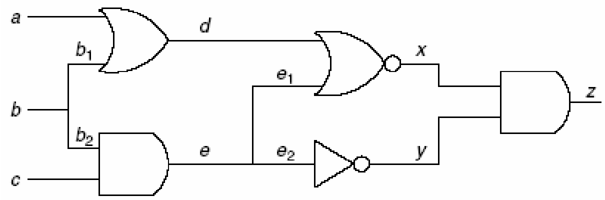
4.6 Untestable Fault Identification – FIRE Example



- Impl[b=1] =
- Faults unexcitable when b=1:
- Faults unobservable when b=1:

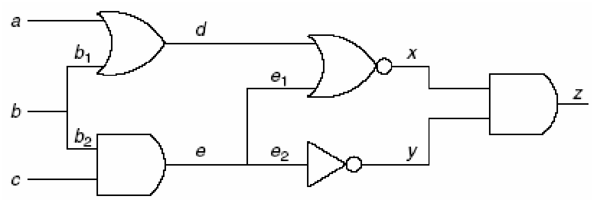
- Fanout stems may still be _____________ even if _______ are not, due to multiple path propagation
- A fanout stem, s, may be observable if both of the following are true
  - s has at least one _______ parity convergence
  - None of the uncontrollable lines involved in blocking are __________ from s

4.6 Untestable Fault Identification – FIRE Example Continued



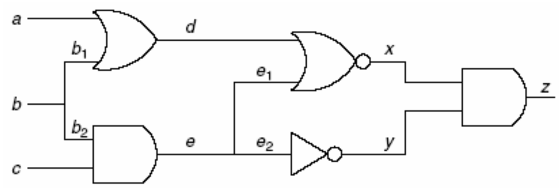
- e satisfies the ______ but not the _______. In this case, add {_________} to the unobservable list and propagate unobservability to include {______________}
- Faults undetectable (union of unexcitable and unobservable) when b=1:

4.6 Untestable Fault Identification – FIRE Example Continued



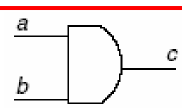
- Impl[b=0] =
- Faults unexcitable when b=0:
- Faults unobservable when b=0:
- Faults undetectable when b=0:

4.6 Untestable Fault Identification – FIRE Example Concluded



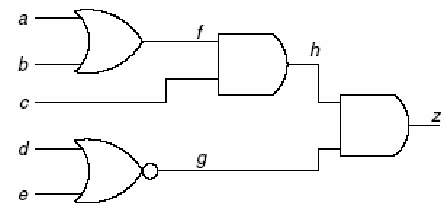
- Now that the two sets of faults undetectable when b=0 and b=1 have been computed
- The _____________ of the two sets are those faults that require b=1 AND b=0 for detection, thus _________:
- b=1: {a/0, a/1, b/1, b_1/1, b_2/0, b_2/1, c/0, c/1, d/1, e/0, e/1, e_1/0, e1/1, e_2/0, e_2/1, x/0, y/0, y/1, z/0}
- b=0: {b/0, b_1/0, b_2/0, c/0, c/1, e_1/0, e_2/0, y/1}
- Intersection and untestable:

4.6 Untestable Faults – Multiple-Line Conflict Analysis



- Consider an AND gate
- {a=0, c=1} is illegal (but this is captured by ________ conflicts)
- Likewise {b=0, c=1}
- But, {a=1, b=1, c=0} is a _________ conflict not captured by __________ conflict
- S_0 – set of faults undetectable when signal a=0
- S_1 – set of faults undetectable when signal b=0
- S_2 – set of faults undetectable when signal c=1
- Intersection of these sets is the set of undetectable faults

4.6 Untestable Faults – Multiple-Line Conflict Analysis(continued)



- Can ________ the previous concept further
- Consider multi-line conflict {______________}
- We can extend these values as far as possible: {____ ________________} is a multi-line conflict as well

4.6 Untestable Faults – Summary

- ❑First compute _____ logic implications
- ❑Compute untestable faults based on ____________ conflicts
- ❑Compute untestable faults based on __________ conflicts
- ❑__________ all identified untestable faults from the fault list

4.10 ATPG for Non-stuck-at Faults – Delay Defects

- Delay defects: class of defects that affects the _______________ only when the circuit is running at __________
- ___________ model insufficient to model all delay-related defects
- Delay fault models
  - _____ delay fault
  - __________ fault
  - __________ delay fault

4.10 ATPG for Non-stuck-at Faults – Delay Defects(Types of Tests)

- Launch on _________ (aka broadside or double capture)
  - V1 is arbitrary, v2 is derived from v1 through the _______ __________
- Launch on ____ (aka skewed load)
  - V1 is arbitrary, v2 is derived by a ____ shift of v1
- _________ ______
  - V1 and V2 are _____________

4.10 ATPG for Non-stuck-at Faults – Delay Defects(Launch on Capture)

- ❑True __________ test
- ❑Benefits
  - ▪Detect intra-clock-domain faults and inter-clock-domain ___________ faults or delay faults at-speed
  - ▪Facilitate _________ implementation
  - ▪_________ some of functionally infeasible paths
  - ▪Ease _________ with ATPG

4.10 ATPG for Non-stuck-at Faults – Delay Defects(Launch on Shift)

- An at-speed delay test technique
- Can address ______________ delay faults
- V1 and V2 correlated
  - ____ ___________ functionally infeasible paths
- Three approaches (details in chapter 5)
  - One-hot skewed-load
  - Aligned skewed-load
  - Staggered skewed-load

4.10 ATPG for Non-stuck-at Faults – Delay Defects(Enhanced Scan)

- Enhanced-scan cells needed
- Larger cells to hold ____ values at each FF
- Can apply two uncorrelated vectors consecutively
  - Can achieve _________ coverage, since all V1-V2 combinations are possible

4.10 ATPG for Non-stuck-at Faults – Classification of Path-Delay Faults

- ❑Models a combinational path in the circuit
  - ▪Considers the ____________ effect of the delay along the path
  - ▪On-inputs of a path
  - ▪Off-inputs of a path
- ❑A ____________ is launched at the start of the path, and a test must propagate the __________ to the end of the path
  - ▪Two faults associated with every path: _____ and _______ transition at the start of the path
- ❑Number of paths can be _______________ to the number of gates in the circuit
- ❑Two vectors needed
  - ▪V1: ___________ vector
  - ▪V2: ______ and __________ vector

4.10 ATPG for Non-stuck-at Faults – Classification of Path-Delay Faults

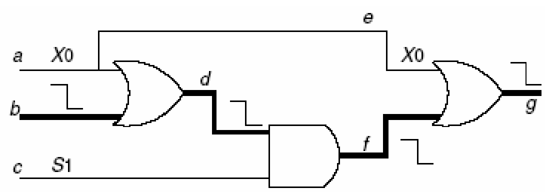
- Statically sensitizable: all ________ of a path P can be assigned to ________________ values by some vector
- Single-path sensitizable: all __________ of a path can be set to _____________ values for both vectors of a test
- ______ path: a transition cannot propagate from the start to the end of path
  - Not all necessary off-input values can be set to non-controlling values ______________

4.10 ATPG for Non-stuck-at Faults – Path-Delay Faults(Robustly Testable)

- If a path is robustly testable, then the corresponding test can verify the correctness of the path _____________ of other _______ in the circuit
- Value criteria for robust testable path:
  - When the corresponding on-input of P has a __________ to _____________ transition, the value in the first vector for the off-input can be __ with the value for the off-input as a _______________ value in the second vector.
  - When the corresponding on-input of P has a ________________ to __________ transition, the values for the off-input must be a ________ non-controlling value for both vectors.

4.10 ATPG for Non-stuck-at Faults – Path-Delay Faults(Robustly Testable)

- Single-path sensitization is too ___________
- May not need to set ____ off-inputs to non-controlling values in V1 in order to propagate a transition
  - ___________ path is robustly testable

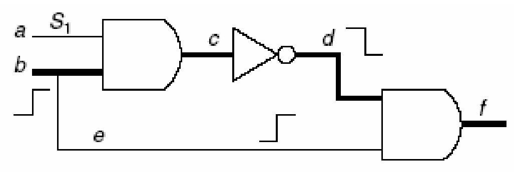


4.10 ATPG for Non-stuck-at Faults – Path-Delay Faults(Nonrobustly Testable)

- Non-robust test only valid if __ ______ delay fault is present in the circuit
- Value criteria for non-robust testing:
  - Irrespective of the __________ on the on-input, the value in the _____ vector for the off-input can be X, with the value for the off-input being a non-controlling value in the _______ vector.

4.10 ATPG for Non-stuck-at Faults – Path-Delay Faults(Nonrobustly Testable)

- Not all paths are __________ testable
- Further _______ requirements for V1
- Test is valid if circuit has __ _____ delay faults
  - Highlighted path is _______________ testable



4.10 ATPG for Non-stuck-at Faults – Path-Delay Faults(Symbols)

- Can use new ________ to consider _____ _______ simultaneously during ATPG
- S0 – Initial and final values are both logic 0
- S1- Initial and final values are both logic 1
- U0 – Initial logic can be either 0 or 1, but final value is logic 0
- U1 – Initial logic can be either 0 or 1, but final value is logic 0
- XX - Both initial and final values are don’t cares

4.10 ATPG for Non-stuck-at Faults – Path-Delay Faults(Boolean Operations)

AND	S0	U0	S1	U1	XX
S0	S0	S0	S0	S0	S0
U0	S0	U0	U0	U0	U0
S1	S0	U0	S1	U1	XX
U1	S0	U0	U1	U1	XX
XX	S0	U0	XX	XX	XX

NOT	
S0	S1
U0	U1
S1	S0
U1	U0
XX	XX

OR	S0	U0	S1	U1	XX
S0	S0	U0	S1	U1	XX
U0	U0	U0	S1	U1	XX
S1	S1	S1	S1	S1	S1
U1	U1	U1	S1	U1	U1
XX	XX	XX	S1	U1	XX

4.10 ATPG for Non-stuck-at Faults – Path-Delay Faults(Transition Fault Model)

- Assumes a __________ delay is present at a circuit node
- _____________ of which path the effect is propagated, the gross delay defect will be late arriving at an ____________ point
- _____ ___________ used in industry
  - ______ and number of faults ______ to circuit size
  - Also needs 2 vectors to test
- Node x slow-to-rise (x-STR) can be modeled simply as two stuck-at faults
  - First time-frame: ___ needs to be ________
  - Second time-frame: ___ needs to be ________ and ____________

4.10 ATPG for Non-stuck-at Faults – Path-Delay Faults(Transition Fault Model ATPG)

- Simply treat each transition fault as two stuck-at faults
- Can test it with _________, __________, or _________ ____
- __________ ____
  - First perform ATPG for _______ faults
  - Then build a __________ for the _______ generated
  - Use the ___________ to identify __________ for each transition fault

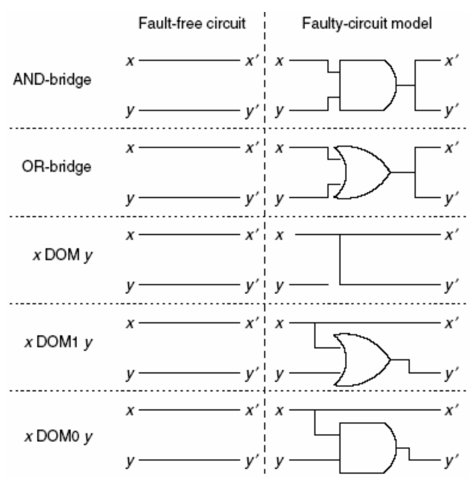
4.10 ATPG for Non-stuck-at Faults – Path-Delay Faults(Transition Test Chains)

Vectors	Excited Faults	Detected Faults
V_1	a/0, b/1, c/1, d/0, e/0	a/0, b/1
V_2	b/1, c/0, d/0, e/1	c/0, d/0, e/1
V_3	a/1, c/1	a/1, c/1
V_4	a/1, b/0, d/1, e/0	b/0, d/1, e/0

4.10 ATPG for Non-stuck-at Faults – Path-Delay Faults(Bridging Fault ATPG)

- Models _______ between two circuit nodes
- The bridge fault is not excited unless the two circuit nodes have _________ logic values
- Faulty value depends on the bridge-fault type:
  - ____ bridge: faulty value is the _____ of the two involved nodes' values
  - ____ bridge: faulty value is the ___ of the two involved nodes' values
  - X Dom y: value of x __________
  - X Dom1 y: x __________ y if ____
  - X Dom0 y: x __________ y if ____

4.10 ATPG for Non-stuck-at Faults – Path-Delay Faults(Bridging Fault ATPG)



- Testing for bridging faults is similar to a __________ stuck-at ATPG.
  - ❑Consider AND-bridge(x,y), we can do either:
    - ▪Detect x/0 with setting y=0
    - ▪Detect y/0 with setting x=0
  - ❑__________ stuck-at ATPG can be modified to handle _____ faults

4.11 Other Topics in Test Generation – Test Compaction

- Want to reduce the _____ _____ _____ to reduce test data _____ and test ___________ time
- First build a detection _________
- __________ vector: a vector that detects some faults that no other vector can detect
  - ____________
- A set _________ algorithm is applied to find a _________ test set such that every fault is covered
- If vectors are ___________ specified
  - Some vectors may be _________: _____ and _____ are __________. Just one vector _____ is sufficient

	f_1	f_2	f_3	f_4	f_5	f_6
v_1	X		X		X	
v_2					X	X
v_3	X			X		X
v_4		X	X	X	X	

4.11 Other Topics in Test Generation – N-Detect ATPG

- Idea: detect every fault at _______ ___ times
  - N vectors that detect a fault must be ___________
- Although the same _____ coverage, can significantly enhance the _______ coverage
  - If x/0 is detected 2 times, one with ____, and the other with ____, then the ____-bridge fault of (x,y) would have been detected by the _______ test
- ATPG can be modified to N-Detect ATPG