Center for Reliable Computing

DUDES:

A Fault Abstraction and Collapsing Framework for Asynchronous Circuits

Philip P. Shirvani
Subhasish Mitra
Jo C. Ebergen[1]
Marly Roncken[2]

Center for Reliable Computing, Stanford University
[1]Sun Microsystems Laboratories
[2]Intel Corporation

Outline

- Motivation and Background
- Our Contributions
- DUDES Fault Model
- Fault Equivalence and Collapsing
- Conclusion

Motivation: Fault Modeling

- Testing for Manufacturing Defects
  - Ensure high quality
- Fault Model Characteristics
  - Effectiveness in detecting defective parts
  - Simplicity
    - Simulation
    - ATPG complexity

Motivation: ATPG Complexity

- Automatic Test Pattern Generation (ATPG)
  - High fault coverage
  - Low test generation time
- Too Many Faults to Cover
  - Prune fault list

Our Contributions

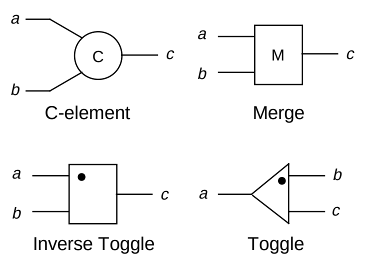
- Fault Modeling
  - Building block inputs/outputs
  - Unified model
  - New faults
    - Pattern-sequence-dependent faults
  - ⋆ DUDES
- Fault Collapsing
  - Basis for a formal technique
- Towards ATPG

Advantages

- Logic-Level
  - + Fast simulation time
- Derived from Transistor Level Stuck-at Faults
  - + Thorough
- Fault Equivalence & Dominance Relationships
  - + Fault list reduction
  - + Speed-up ATPG
- Uniform Model
  - + Hierarchical approach
  - + Applicable at system level

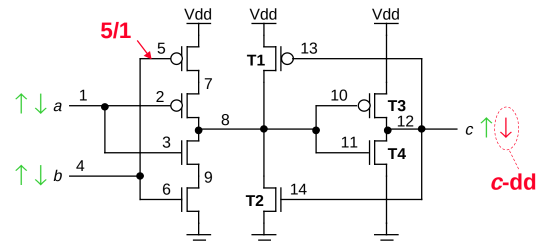
These slides were prepared to accompany "DUDES: A Fault Abstraction and Collapsing Framework for Asynchronous Circuits," published in *Proceedings of the Sixth International Symposiumon on Advanced Research in Asynchronous Circuits and Systems*, Eilat, Israel, 4-6 Israel, 4-6 April 2000. pp. 73-82.

Basic Elements



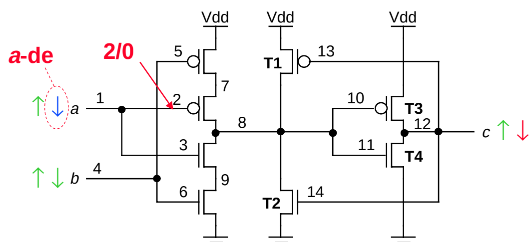
Down/Up-Disabled Output Faults

- Prohibited Transition
  - $a\uparrow$; $b\uparrow$; $c\uparrow$; $a\downarrow$; $b\downarrow$; $c\downarrow$; $a\uparrow$; $b\uparrow$; $c\uparrow$; ...
  - $a\uparrow$; $b\uparrow$; $c\uparrow$; $a\downarrow$; $b\downarrow$; X ; X ; X ; X ; X ; ...



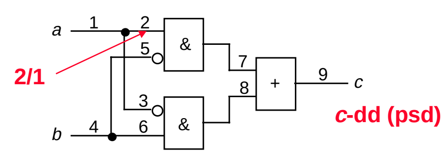
Down/Up-Enabled Input Faults

- Premature Firing
  - $a\uparrow$; $b\uparrow$; $c\uparrow$; $b\downarrow$; $a\downarrow$; $c\downarrow$; ...
  - $a\uparrow$; $b\uparrow$; $c\uparrow$; $b\downarrow$; $c\downarrow$; ...



Pattern Sequence Dependency

- PSD Prohibited Transition (Merge)
  - Initial state *abc* = 101
  - $b\uparrow$; $c\downarrow$; $a\downarrow$; $c\uparrow$; ...
  - $a\downarrow$; X ; X ; X ; X ; X ; ...
  - PSD down-disabled fault on *c*

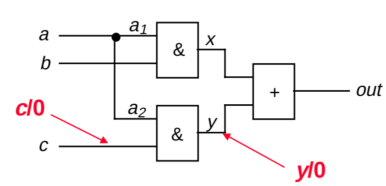


DUDES Fault Model

- Down/Up-Disabled/Enabled and Stuck-at
- Basic Elements (Inputs *x*, *y*, Output *z*)
- Disabled Faults
  - *z* up-disabled, *z* down-disabled
- Enabled Faults
  - *x* up-enabled, *x* down-enabled
  - *y* up-enabled, *y* down-enabled
- Stuck-at Faults
  - *x*/0, *x*/1, *y*/0, *y*/1, *z*/0, *z*/1
- Pattern Sequence Dependency

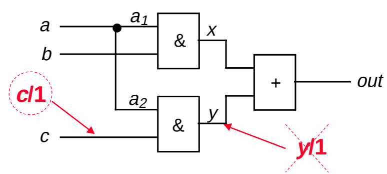
Fault Equivalence

- Example
  - *c*/0 : *out* = *ab*
  - *y*/0 : *out* = *ab*
- Keep Only One in Fault List



Fault Dominance

- Test Set Comparison
  - $y/1$: A = { abc = 000, 001, 010, 011, 100 }
  - $c/1$: B = { abc = 100 }
  - B $\subseteq$ A $\Rightarrow$ $c/1$ dominated by $y/1$
- Keep Only the Dominated in Fault List ($c/1$)



Fault Collapsing 1: C-Element

- Up/Down-Disabled Output Faults
  - Test sequence = $(a\uparrow;\ b\uparrow;\ c\uparrow;\ a\downarrow;\ b\downarrow;\ c\downarrow)$
- Pin Stuck-at Faults
  - Same test sequence
- Up/Down-Enabled Input Faults
  - ($a\uparrow$; <w>; $b\uparrow$; $c\uparrow$; $a\downarrow$; <w>; $b\downarrow$; $c\downarrow$;
    $b\uparrow$; <w>; $a\uparrow$; $c\uparrow$; $b\downarrow$; <w>; $a\downarrow$; $c\downarrow$)
  - Dominated by disabled and stuck-at faults
- Keep Only the Up/Down-Enabled Input Faults

Fault Collapsing 2: Inverse Toggle

- $pref$ * $[a\uparrow;\ c\uparrow;\ b\uparrow;\ c\downarrow;\ a\downarrow;\ c\uparrow;\ b\downarrow;\ c\downarrow]$
- Initial state abc = 000
- Test for $a/0$ and $c/0$ is $(a\uparrow;\ c\uparrow)$
  - $a/0$ and $c/0$ equivalent
- Test for $b/0$ is $(a\uparrow;\ c\uparrow;\ b\uparrow;\ c\downarrow)$
  - $b/0$ dominated by $a/0$ and $c/0$
  - Keep only $b/0$

Formalizing Fault Collapsing

- Unrolled Trace Expressions
  - Transition directions
  - Disjunction-free subexpressions
- C-Element

  C(x, y, z) = $pref$ * [(x; y) | (y; x) ; z]

  $\Rightarrow$

  $$C(x, y, z) = pref\ *\ [\ (x\uparrow;\ y\uparrow;\ z\uparrow;\ x\downarrow;\ y\downarrow;\ z\downarrow)$$
  $$|\ (x\uparrow;\ y\uparrow;\ z\uparrow;\ y\downarrow;\ x\downarrow;\ z\downarrow)$$
  $$|\ (y\uparrow;\ x\uparrow;\ z\uparrow;\ x\downarrow;\ y\downarrow;\ z\downarrow)$$
  $$|\ (y\uparrow;\ x\uparrow;\ z\uparrow;\ y\downarrow;\ x\downarrow;\ z\downarrow)\ ]$$

Fault Collapsing 1: Stuck-At Faults

- $x/0$ in C-Element(x, y, z)
  - no transition on $x \Rightarrow$ no transition on z
  - Remove $z\uparrow$ & $z\downarrow$ and following events

  $$C(x, y, z) = pref\ *\ [\ (x\uparrow;\ y\uparrow;\ z\uparrow;\ x\downarrow;\ y\downarrow;\ z\downarrow)$$
  $$|\ (x\uparrow;\ y\uparrow;\ z\uparrow;\ y\downarrow;\ x\downarrow;\ z\downarrow)$$
  $$|\ (y\uparrow;\ x\uparrow;\ z\uparrow;\ x\downarrow;\ y\downarrow;\ z\downarrow)$$
  $$|\ (y\uparrow;\ x\uparrow;\ z\uparrow;\ y\downarrow;\ x\downarrow;\ z\downarrow)\ ]$$

  $$\Rightarrow\quad pref\ [\ (x\uparrow;\ y\uparrow)\ |\ (y\uparrow;\ x\uparrow)\ ]$$
- Same Expression for $y/0$ and $z/0$
  - $x/0$, $y/0$ and $z/0$ are equivalent
  - Keep only one of them

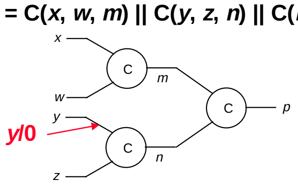
Fault Collapsing 2: Up-Enabled Faults

- x Up-Enabled in C-Element(x, y, z)
  - $x\uparrow$ is already satisfied
  - Remove $x\uparrow$ from sub-expressions
- Need $x\uparrow$ Before $x\downarrow$
  - Weave with $pref$ * [$x\uparrow$; $x\downarrow$]

  $$pref\ *\ [\ (y\uparrow;\ z\uparrow;\ x\downarrow;\ y\downarrow;\ z\downarrow)$$
  $$|\ (y\uparrow;\ z\uparrow;\ y\downarrow;\ x\downarrow;\ z\downarrow)\ ]$$
  $$\|\ \ pref\ *\ [\ x\uparrow;\ x\downarrow\ ]$$

System-Level Fault Collapsing (1)

F = C(x, w, m) || C(y, z, n) || C(m, n, p)

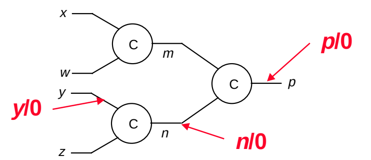


- C(x, w, m) || *pref* [($y\uparrow$; $z\uparrow$) | ($z\uparrow$; $y\uparrow$)] || C(m, n, p)
- Final expr.: one occurrence of $x\uparrow$, $w\uparrow$, $y\uparrow$, $z\uparrow$
- Same for w/0
  - y/0 and w/0 are equivalent

DUDES - Async 2000 19

System-Level Fault Collapsing (2)

- Equivalence Relationship
  - From low-level collapsing
  - To high-level collapsing



y/0 ≡ n/0 and n/0 ≡ p/0 ⇒ y/0 ≡ p/0

DUDES - Async 2000 20

Conclusions

- DUDES Fault Model
  - Map internal stuck-at faults to pin faults
  - PSD and <w> attribute used for ATPG
- Supports Hierarchical Analysis at Logic Level
  - Very effective for fault collapsing
    - From basic elements
    - To system level
- Work in Progress
  - Formalization
  - ATPG algorithms

DUDES - Async 2000 21

DUDES - Async 2000 22

Previous Work

- Stuck Faults on Building Block Inputs/Outputs
  - \+ Fast simulation time
  - \+ Self-checking properties
  - – Not thorough enough
- Very High-Level Fault Modeling
  - – Effectiveness not known
- Transistor-Level Stuck-at Faults
  - \+ Thorough
  - – Mainly for C-elements
  - – Long simulation time

DUDES - Async 2000 23