Intelligent Targeting of Cell-Aware-Faults by the use of mandatory conditions

Micah A. Thornton
mathornton@smu.edu

Fanchen Zhang
fzhang@smu.edu

Jennifer Dworak
jdworak@smu.edu

Southern Methodist University
Department of Computer Science and Engineering

Outline

1. Introduction
   - Cell-Aware-Type Faults
   - Cell Aware Type Example
   - Functional Simulation
   - Motivation

Outline

1. Introduction
   - Cell-Aware-Type Faults
   - Cell Aware Type Example
   - Functional Simulation
   - Motivation
2. Experiment
   - Cell-Aware-Type UDFM Generation
   - Mandatory Condition Extraction
   - Circuit Goodstate Extraction Functional Simulation
   - Mandatory Counts During Functional Simulation

Outline

Outline

1. Introduction
   - Cell-Aware-Type Faults
   - Cell Aware Type Example
   - Functional Simulation
   - Motivation
2. Experiment
   - Cell-Aware-Type UDFM Generation
   - Mandatory Condition Extraction
   - Circuit Goodstate Extraction Functional Simulation
   - Mandatory Counts During Functional Simulation
3. Results & Analysis
   - Mandatory Counts as Fault Classifiers
   - Regression Analysis
4. Conclusion

Outline

Previous Work

Previous Work

- Cell-Aware Fault Model

Previous Work

- Cell-Aware Fault Model
  - Proposed by Hapke et. al

Previous Work

- Cell-Aware Fault Model
  - Proposed by Hapke et. al
  - Models Analog Faults

Previous Work

- Cell-Aware Fault Model
  - Proposed by Hapke et. al
  - Models Analog Faults
  - Difficult to do ATPG

Previous Work

- Cell-Aware Fault Model
  - Proposed by Hapke et. al
  - Models Analog Faults
  - Difficult to do ATPG
  - Models faults as occurring within standard cells (logic gates)

Previous Work

- Cell-Aware Fault Model
  - Proposed by Hapke et. al
  - Models Analog Faults
  - Difficult to do ATPG
  - Models faults as occurring within standard cells (logic gates)
  - different from "Cell-Aware-Type." (More on this to come)

Previous Work

- Cell-Aware Fault Model
  - Proposed by Hapke et. al
  - Models Analog Faults
  - Difficult to do ATPG
  - Models faults as occurring within standard cells (logic gates)
  - different from "Cell-Aware-Type." (More on this to come)
- Functional Simulation

Previous Work

- Cell-Aware Fault Model
  - Proposed by Hapke et. al
  - Models Analog Faults
  - Difficult to do ATPG
  - Models faults as occurring within standard cells (logic gates)
  - different from "Cell-Aware-Type." (More on this to come)
- Functional Simulation
  - Used to Determine Fault Coverage (Shi '11)

Previous Work

- Cell-Aware Fault Model
  - Proposed by Hapke et. al
  - Models Analog Faults
  - Difficult to do ATPG
  - Models faults as occurring within standard cells (logic gates)
  - different from "Cell-Aware-Type." (More on this to come)
- Functional Simulation
  - Used to Determine Fault Coverage (Shi '11)
  - Overview valid state space

Previous Work

- Cell-Aware Fault Model
  - Proposed by Hapke et. al
  - Models Analog Faults
  - Difficult to do ATPG
  - Models faults as occurring within standard cells (logic gates)
  - different from "Cell-Aware-Type." (More on this to come)
- Functional Simulation
  - Used to Determine Fault Coverage (Shi '11)
  - Overview valid state space
- Extension of work on targeting very difficult stuck-at faults

CAT Faults

CAT Faults

- Cell-Aware Type faults

CAT Faults

- Cell-Aware Type faults
  - Discussed in Paper by Zhang et. al

CAT Faults

- Cell-Aware Type faults
  - Discussed in Paper by Zhang et. al
  - Model Cell-Aware Faults

CAT Faults

- Cell-Aware Type faults
  - Discussed in Paper by Zhang et. al
  - Model Cell-Aware Faults
  - Use Stuck-At-ATPG to differentiate cell-aware-type faults.

CAT Faults

- Cell-Aware Type faults
  - Discussed in Paper by Zhang et. al
  - Model Cell-Aware Faults
  - Use Stuck-At-ATPG to differentiate cell-aware-type faults.

Note

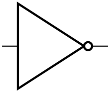
This type of fault is considered different than a pure cell-aware fault because no Analog analysis is required to generate tests. This will be illustrated during the next example

CAT Example

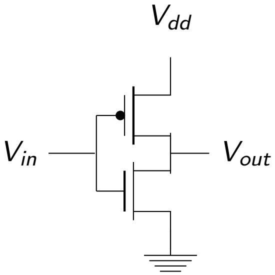
CAT Example

imagine you have an inverter...

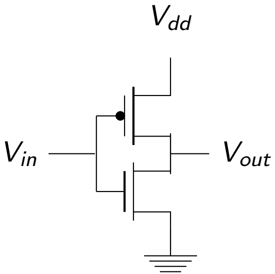
CAT Example



CAT Example



CAT Example



Transistor Parameters

$V_{tn} = V_{tp} = 0.7$ Volts.

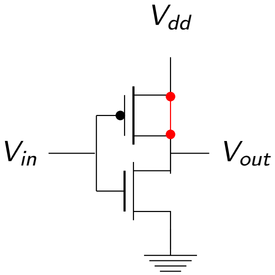
CAT Example

But Transistors are not linear elements, so our model is still not perfect...

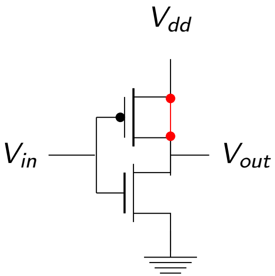
CAT Example

During the manufacture of this inverter...

CAT Example

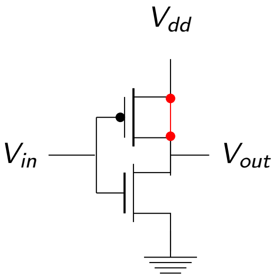


CAT Example



In this simple example cell...

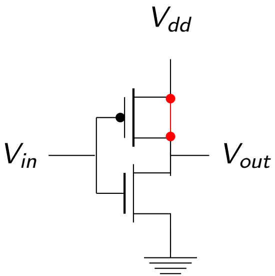
CAT Example



In this simple example cell...

Set $V_{in} = 1$, and observe $V_{out} = 1$

CAT Example



In this simple example cell...

Set $V_{in} = 1$, and observe $V_{out} = 1$

This analysis is difficult to perform on millions of transistors

CAT Example

With Cell-Aware-Type faults, we examine stuck-at ATPG test patterns...

CAT Example

With Cell-Aware-Type faults, we examine stuck-at ATPG test patterns...
And add patterns that cause conflict, but might not be choosen by ATPG tool.

Functional Simulation

Recall that the state space of an automaton refers to...

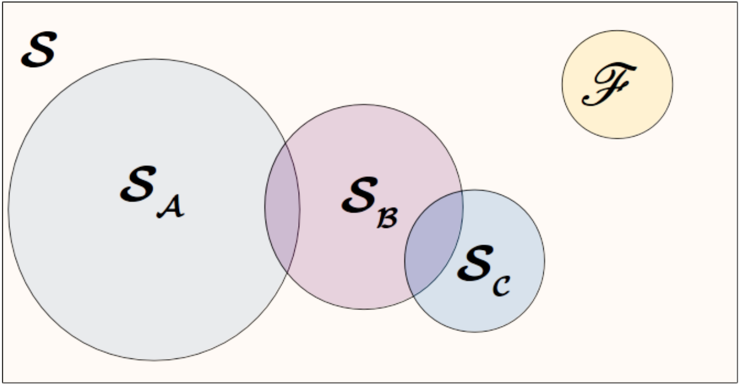
Functional Simulation

Recall that the state space of an automaton refers to...
All possible configurations of the memory elements in a device.

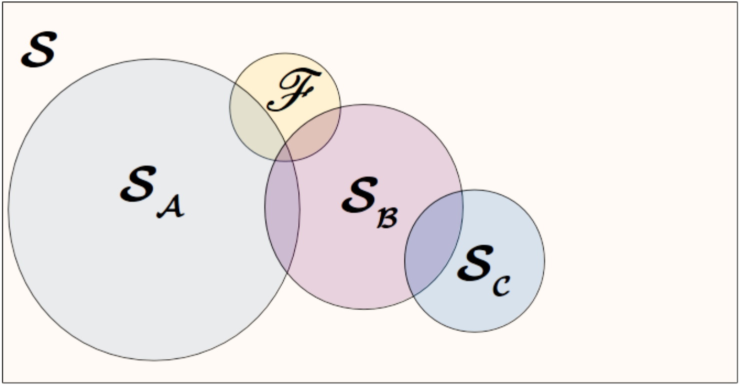
Functional Simulation

Recall that the state space of an automaton refers to...
All possible configurations of the memory elements in a device.
Consider a general purpose device that contains fault F as shown:

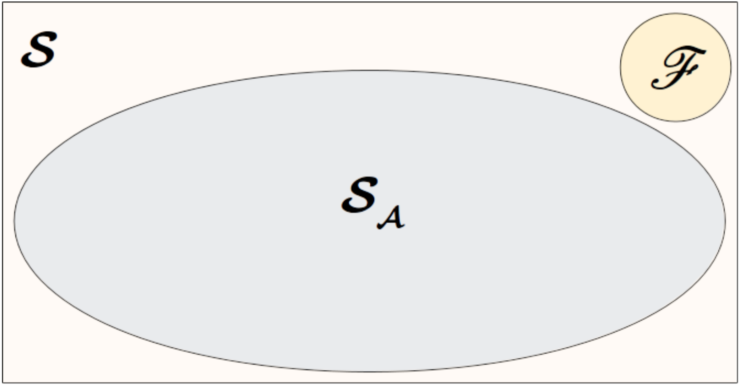
Functional Simulation



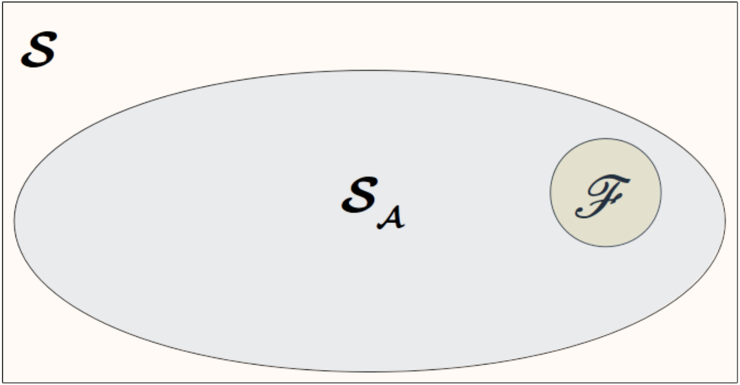
Functional Simulation



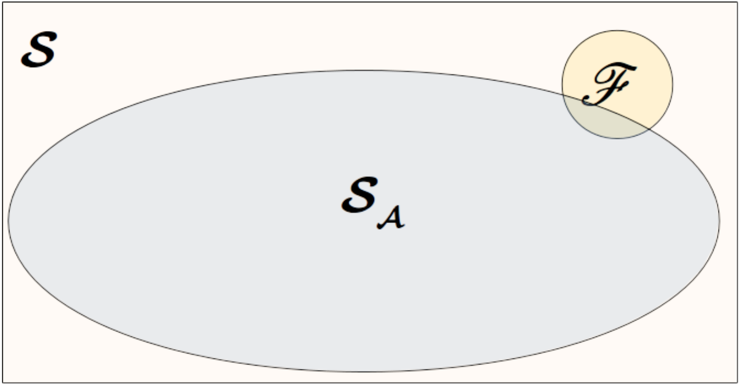
Functional Simulation



Functional Simulation



Functional Simulation



CAT Example

ATPG for pure Cell-Aware Faults is Hard...

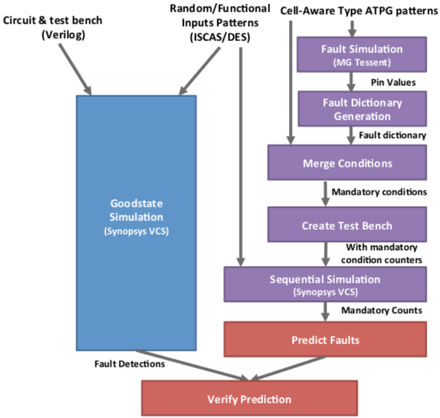
CAT Example

ATPG for pure Cell-Aware Faults is Hard...
It requires many resources (time/computational power)

CAT Example

ATPG for pure Cell-Aware Faults is Hard...
It requires many resources (time/computational power)
Let's prioritize faults using functional analysis of faults.

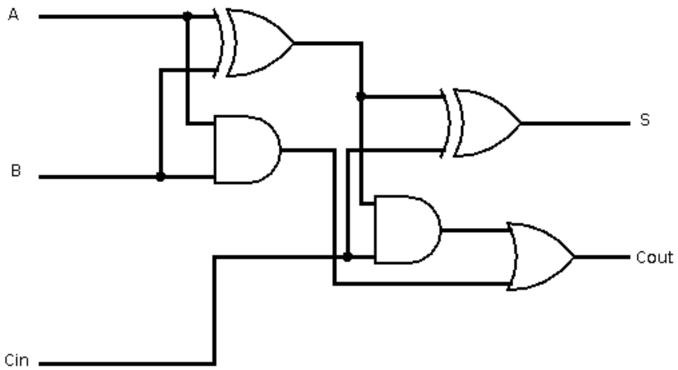
Flow Chart


Circuit & test bench
(Verilog)
Random/Functional
Inputs Patterns
(ISCAS/DES)
Cell-Aware Type ATPG patterns
Fault Simulation
(MG Tessent)
Pin Values
Fault Dictionary
Generation
Fault dictionary
Merge Conditions
Mandatory conditions
Create Test Bench
With mandatory
condition counters
Goodstate
Simulation
(Synopsys VCS)
Sequential Simulation
(Synopsys VCS)
Mandatory Counts
Predict Faults
Fault Detections
Verify Prediction

Cell-Aware-Type UDFM Generation

```
model s_faddx1_CO_(CO, CI, B, A)(
  model_source = verilog_udp;
  input (CI) ( )
  input (B) ( )
  input (A) ( )
  output (CO) (    )
  (
    primitive = _and mlc_sop_product_gate0 (B, A,
        mlc_product_net0_0);
    primitive = _and mlc_sop_product_gate1 (CI, A,
        mlc_product_net0_1);
    primitive = _and mlc_sop_product_gate2 (CI, B,
        mlc_product_net0_2);
    primitive = _or mlc_sop_sum_gate0 (mlc_product_net0_0,
        mlc_product_net0_1, mlc_product_net0_2, CO);
  ))
```

Cell-Aware-Type UDFM Generation



Cell-Aware-Type UDFM Generation

A	B	C_{in}	C_{out}	S
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

Cell-Aware-Type UDFM Generation

```
Cell ( "FADDX1" ) {
    Fault ( "FADDX1_i000_o1" ){
        Test {
                StaticFault { "S" : 1; }
                Conditions { "A" : 0; "B": 0; "CI":
                    0;}
                        }
                }
    Fault ( "FADDX1_i000_o1" ) {
        Test {
                StaticFault { "CO" : 1; }
                Conditions { "A" : 0; "B": 0; "CI":
                    0;}
                        }
                }
        }
```

Mandatory Condition Extraction

Imagine you are testing a circuit with...

Mandatory Condition Extraction

Imagine you are testing a circuit with...
6 Primary Inputs

Mandatory Condition Extraction

Imagine you are testing a circuit with...
6 Primary Inputs
2 State Elements

Mandatory Condition Extraction

Imagine you are testing a circuit with...
6 Primary Inputs
2 State Elements
you are looking for a fault f

Mandatory Condition Extraction

Imagine you are testing a circuit with...
6 Primary Inputs
2 State Elements
you are looking for a fault f
and perform stuck-at-ATPG 4 times (or with n=4 on n-detect)

Mandatory Condition Extraction

These were the patterns that detected it...

Mandatory Condition Extraction

These were the patterns that detected it...

	Inputs	Flip-Flops
Pattern 1	010111	00
Pattern 2	001001	10
Pattern 3	011111	00
Pattern 4	000001	10

Mandatory Condition Extraction

These were the patterns that detected it...

	Inputs	Flip-Flops
Pattern 1	010111	00
Pattern 2	001001	10
Pattern 3	011111	00
Pattern 4	000001	10

$$MC(f) =$$

Mandatory Condition Extraction

These were the patterns that detected it...

	Inputs	Flip-Flops
Pattern 1	010111	00
Pattern 2	001001	10
Pattern 3	011111	00
Pattern 4	000001	10

$$MC(f) = \overline{p_0}$$

Mandatory Condition Extraction

These were the patterns that detected it...

	Inputs	Flip-Flops
Pattern 1	010111	00
Pattern 2	001001	10
Pattern 3	011111	00
Pattern 4	000001	10

$$MC(f) = \overline{p_0} p_5$$

Mandatory Condition Extraction

These were the patterns that detected it...

	Inputs	Flip-Flops
Pattern 1	010111	00
Pattern 2	001001	10
Pattern 3	011111	00
Pattern 4	000001	10

$$MC(f) = \overline{p_0} p_5 \overline{d_1}$$

Circuit Goodstate Extraction Functional Simulation

Circuit Goodstate Extraction Functional Simulation

- This was reported on in the last publication.

Circuit Goodstate Extraction Functional Simulation

- This was reported on in the last publication.
- we inserted scanchains into circuits (ISCAS/DES56)

Circuit Goodstate Extraction Functional Simulation

- This was reported on in the last publication.
- we inserted scanchains into circuits (ISCAS/DES56)
- Random inputs for ISCAS circuits

Circuit Goodstate Extraction Functional Simulation

- This was reported on in the last publication.
- we inserted scanchains into circuits (ISCAS/DES56)
- Random inputs for ISCAS circuits
- Functional Testbench patterns for DES (both encryption and decryption)

Circuit Goodstate Extraction Functional Simulation

- This was reported on in the last publication.
- we inserted scanchains into circuits (ISCAS/DES56)
- Random inputs for ISCAS circuits
- Functional Testbench patterns for DES (both encryption and decryption)
- Captured state after every clock cycle, and had functional states for circuits.

Mandatory Counts During Functional Simulation

Mandatory Counts During Functional Simulation

- After determining the mandatory conditions for each circuit

Mandatory Counts During Functional Simulation

- After determining the mandatory conditions for each circuit
- Mandatory-Condition checking and gates were added to each circuit

Mandatory Counts During Functional Simulation

- After determining the mandatory conditions for each circuit
- Mandatory-Condition checking and gates were added to each circuit
- Using the goodstates we extracted

Mandatory Counts During Functional Simulation

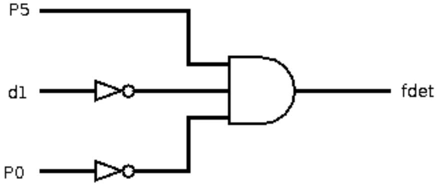
- After determining the mandatory conditions for each circuit
- Mandatory-Condition checking and gates were added to each circuit
- Using the goodstates we extracted
- We performed functional simulation on the circuit,

Mandatory Counts During Functional Simulation

- After determining the mandatory conditions for each circuit
- Mandatory-Condition checking and gates were added to each circuit
- Using the goodstates we extracted
- We performed functional simulation on the circuit,
- and counted the number of times the mandatory conditions occurred.

MAND gates

Figure : Mandatory Condition Detector for fault f



ISCAS s9234

		Detected	
		T	F
Predicted	T	453 (TP)	119 (FP)
	F	0 (FN)	462 (TN)

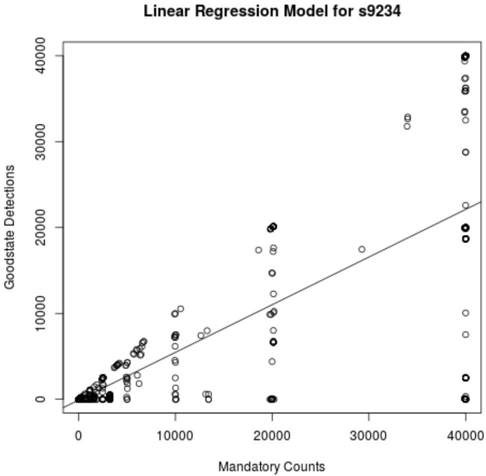
Statistic	Value
Precision	79%
Accuracy	88%
Specificity	79%
Fall-out	20.5%

DES 56

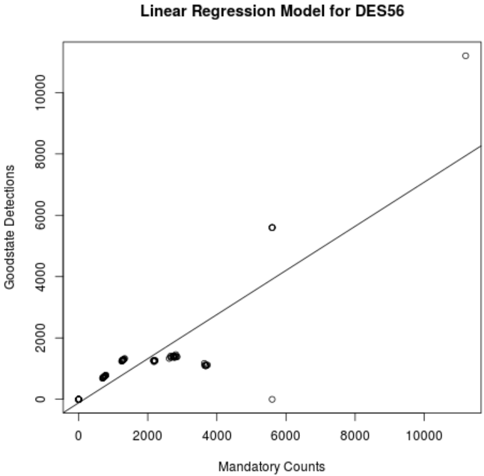
		Detected	
		T	F
Predicted	T	461 (TP)	2 (FP)
	F	0 (FN)	14 (TN)

Statistic	Value
Sensitivity	100%
Accuracy	99.5%
Specificity	87.5%
Fall-out	14.2%
Precision	99.5%

ISCAS s9234


Linear Regression Model for s9234
Goodstate Detections
0
10000
20000
30000
40000
0
10000
20000
30000
40000
Mandatory Counts

DES 56


Linear Regression Model for DES56
Goodstate Detections
0
2000
4000
6000
8000
10000
0
2000
4000
6000
8000
10000
Mandatory Counts

Conclusion

Conclusion

- Cell-Aware fault model describes faults within cells

Conclusion

- Cell-Aware fault model describes faults within cells
- Large number of C.A. Faults for given circuits

Conclusion

- Cell-Aware fault model describes faults within cells
- Large number of C.A. Faults for given circuits
- Functional Simulation and Mandatory Conditions allow us to prioritize fault detections

Conclusion

- Cell-Aware fault model describes faults within cells
- Large number of C.A. Faults for given circuits
- Functional Simulation and Mandatory Conditions allow us to prioritize fault detections
- We provided examples of mandatory condition calculations, and showed how they could be used to predict whether or not a cell-aware fault is functional

Acknowledgement

The authors would like to thank Semiconductor Research Corporation who supports this research under Task ID 2465.001. The authors are also grateful for the support of our industrial liaisons.

Thank You

Questions?