Carnegie Mellon

Yield Learning in 7nm Semiconductor Technologies: Test Chip Design

SHAWN BLANTON

Advanced Chip Testing Laboratory

ECE Department

Carnegie Mellon University

www.ece.cmu.edu/~actl

January 5, 2020

© R. D. Blanton

Advanced Chip Testing Laboratory (ACTL)

Hardware Security

Trojans

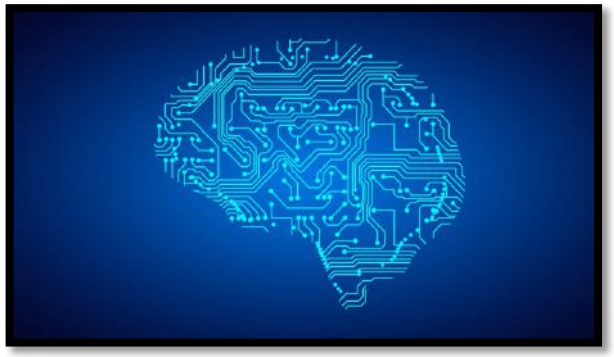
Counterfeiting

Reverse engineering

Obfuscation

Logic locking

Machine Learning



Security

Computing

Design Automation

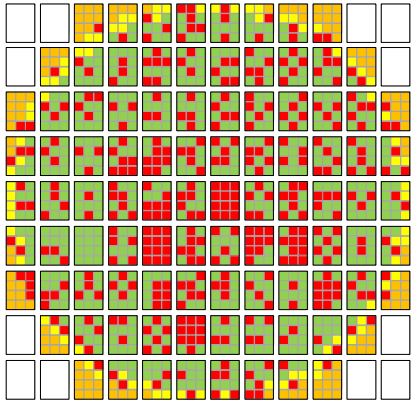
Test

Diagnosis

Yield

Hardware Acceleration

Yield Learning



Diagnosis

Data Analytics

Test Chip Design

Typical Process Development Cycle

- Fast yield improvement is crucial for new manufacturing process
- Test chip is one tool for yield learning during the process development cycle

	Stage I: Technology exploration	Stage II: Module definition	Stage III: Process validation	Stage IV: Technology validation	Stage V: Process yield ramp	Stage VI: Product yield ramp
Test chips	Proof-of-technology	Proof-of-module	Single-layer structures	Short-flow FEOL/BEOL	Short-flow FEOL/BEOL	Product
		Transistor structures	Std. cell structures	Full-flow small SAPR	Full-flow large SAPR	Full-flow SAPR
		SRAM (2Mb)	SRAM (2Mb)	SRAM (2Mb)	SRAM (2Mb)	
PDK version	NA	0.01	0.1	0.5	1.0	

Yield Learning with Test Strucutres

◊ **Process development requires test structures that:**

- **Range from simple wires to product-like designs**
- **Are transparent to failure**

◊ **Logic is the most difficult to yield**

◊ **Logic characterization vehicle (LCV) design is ad hoc**

- **Typically a sub-circuit from a legacy design**
- **Lacks sufficient transparency to failure**

OBJECTIVE: A design methodology for LCVs that are reflective of actual designs and failure transparent

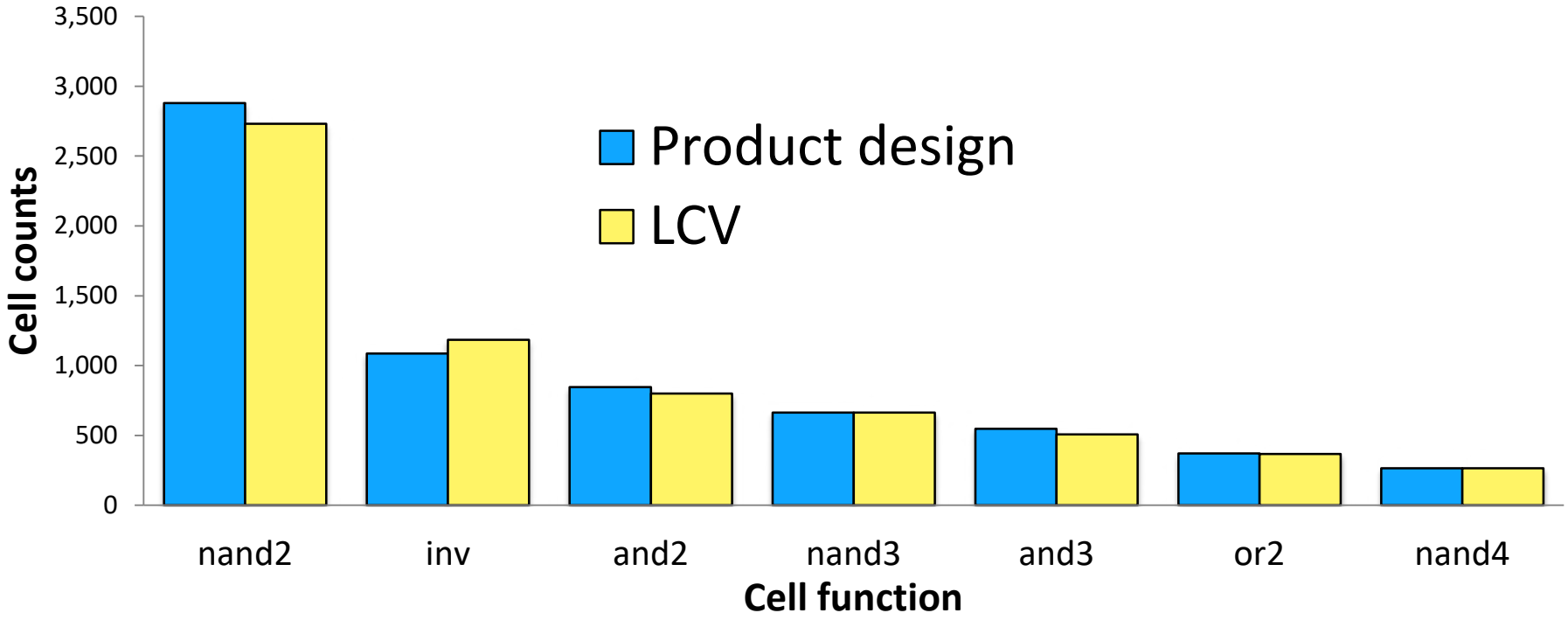
Transparency and Reflection

◊ **Transparency** ≡ **Circuit failure easy to characterize**

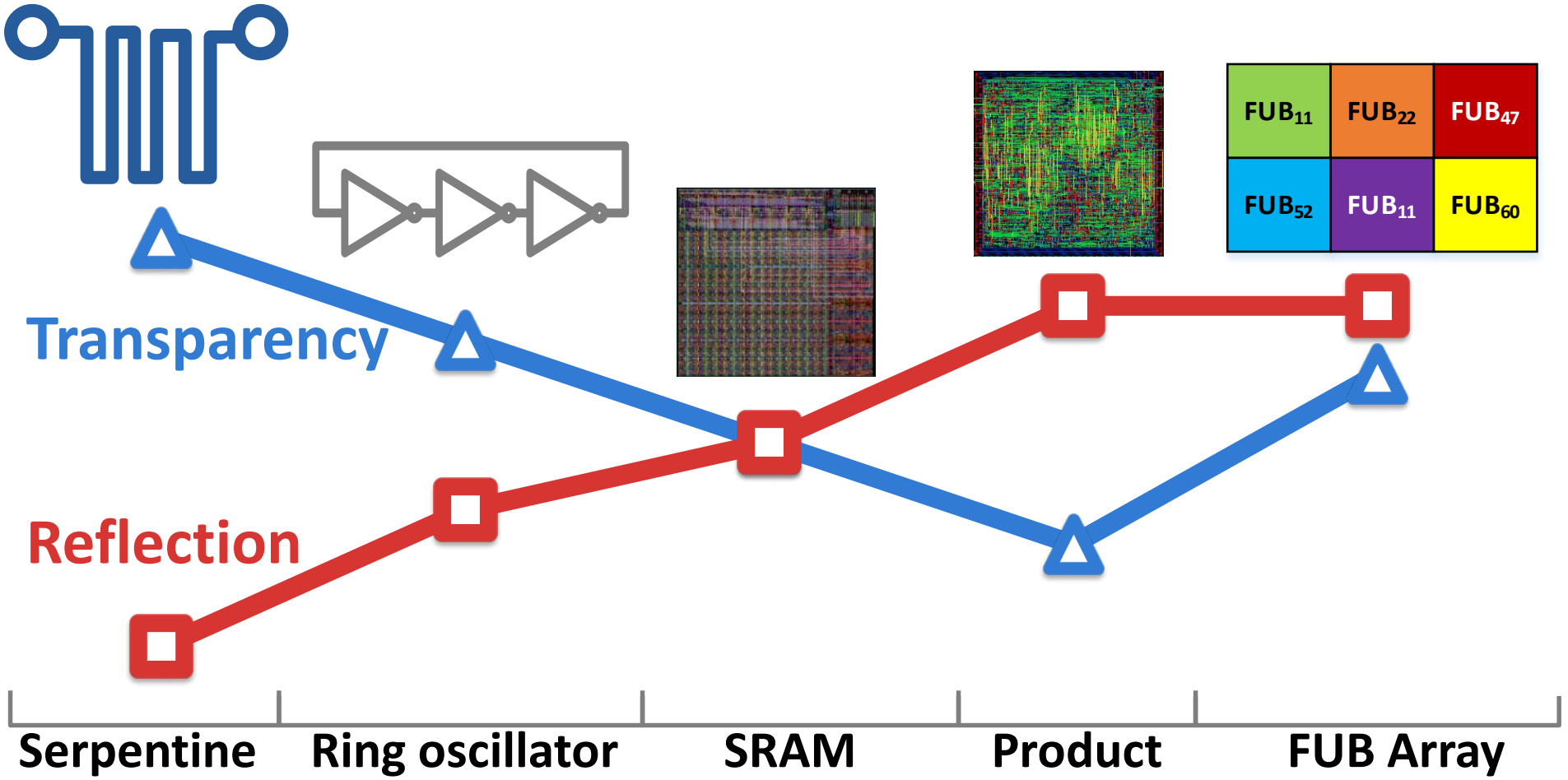
- **That is, testable, diagnosable**

◊ **Reflection** ≡ **LCV has demographics of real designs**

- **E.g., contains the same standard cells**



Reflective Design .vs. Transparency

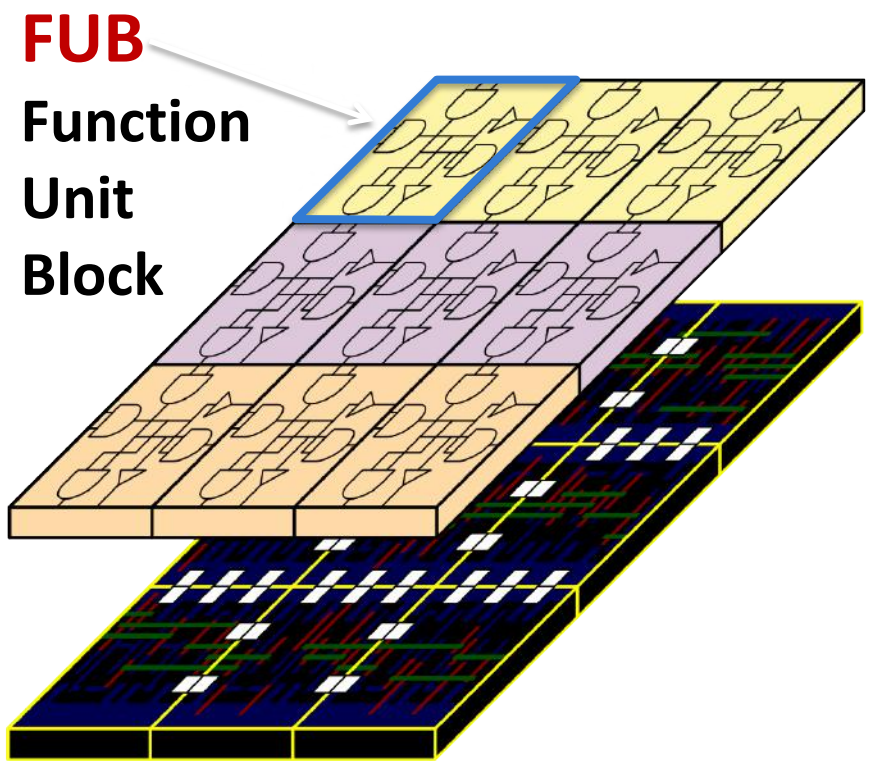


Industry Survey

LCV Objectives and Needs	Score
1. Identify Troublesome Layout Patterns	11
2. Evaluate Place and Route	11
3. Improved Defect Characterization	10
4. Improved Testability	9
5. Investigate Known Yield Issues	9
6. BEOL Design Rules	8
7. BEOL DFM Guidelines	8
8. Neighborhood Effects	8
9. Standard Cell Evaluation	**7**
10. FEOL Design Rules	6
11. FEOL DFM Guidelines	6
12. Reduced Test Time	6
13. Increased Density	5
14. Reduced Cost	5
15. Verification	3

Insight: Logic and layout are separable

- Logic function, structure → test, diagnosis
- Layout→ Libraries, DFM, place-&-route, etc.



Logic design and structure

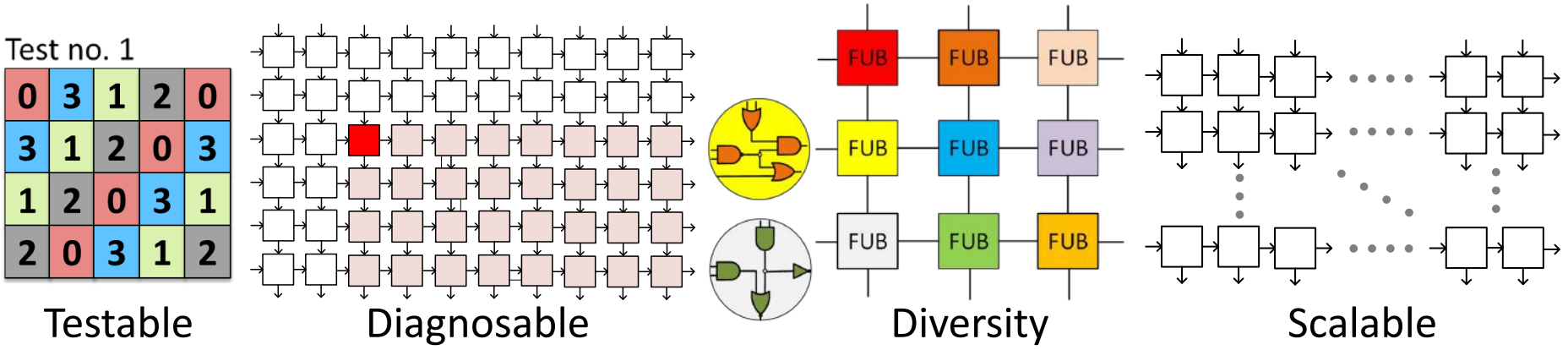
- 2D logic array of FUBs
- Optimal test and diagnosis
- Maximal design density
- High fault coverage of every FUB

Layout design

- Virtually independent of logic function
- Custom, synthesis + P&R, etc.
- FEOL, MEOL and/or BEOL emphasis
- Reflect properties of actual products

CM-LCV Architecture: 2D FUB Array

- ◊ **Each FUB implements a bijective function**
- ◊ **Bijectivity guarantees constant-testability**
  - **C-testable:** Each FUB is fully testable with min. tests
  - **Diagnosable:** Error outputs intersect with defective FUB
  - **Diversity:** Properties hold regardless of FUB logic
  - **Scalable:** Properties hold regardless or array size



FUB Array C-Testability

2D array of bijective FUBs is C-testable*

- FUBs are exhaustively tested with min. no. of tests
- True regardless of array size
- Each IP fault (truth table row) is tested (verified)

Bijective function

In	Out
0	3
1	2
2	0
3	1

Test no. 1

2	0	3	1	2
0	3	1	2	0
3	1	2	0	3
1	2	0	3	1

Test no. 2

1	2	0	3	1
2	0	3	1	2
0	3	1	2	0
3	1	2	0	3

Test no. 3

3	1	2	0	3
1	2	0	3	1
2	0	3	1	2
0	3	1	2	0

Test no. 4

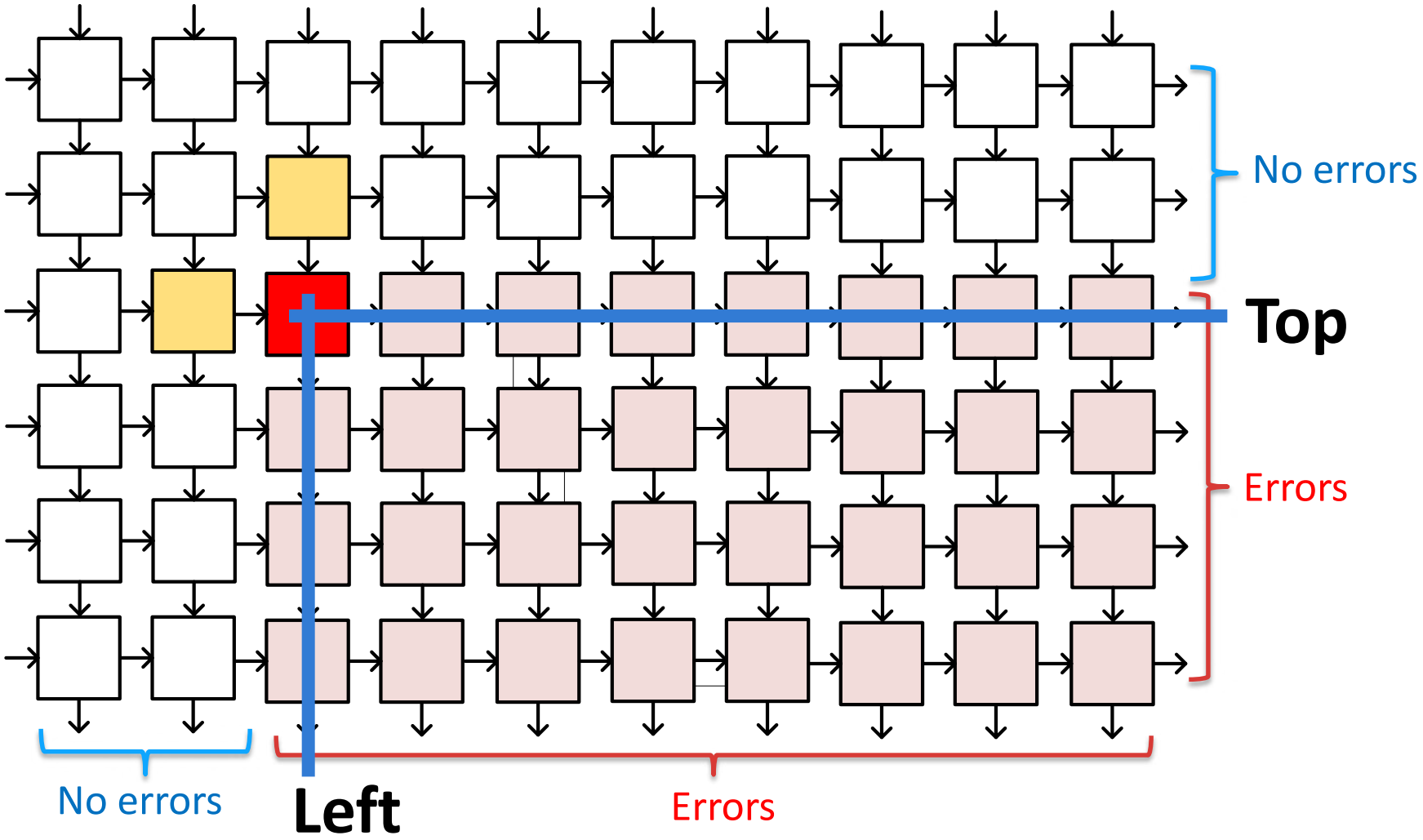
0	3	1	2	0
3	1	2	0	3
1	2	0	3	1
2	0	3	1	2

Each test is just a rotated version of the previous test

*A. D. Friedman, “Easily Testable Iterative Systems,” IEEE TCAD, 1973.

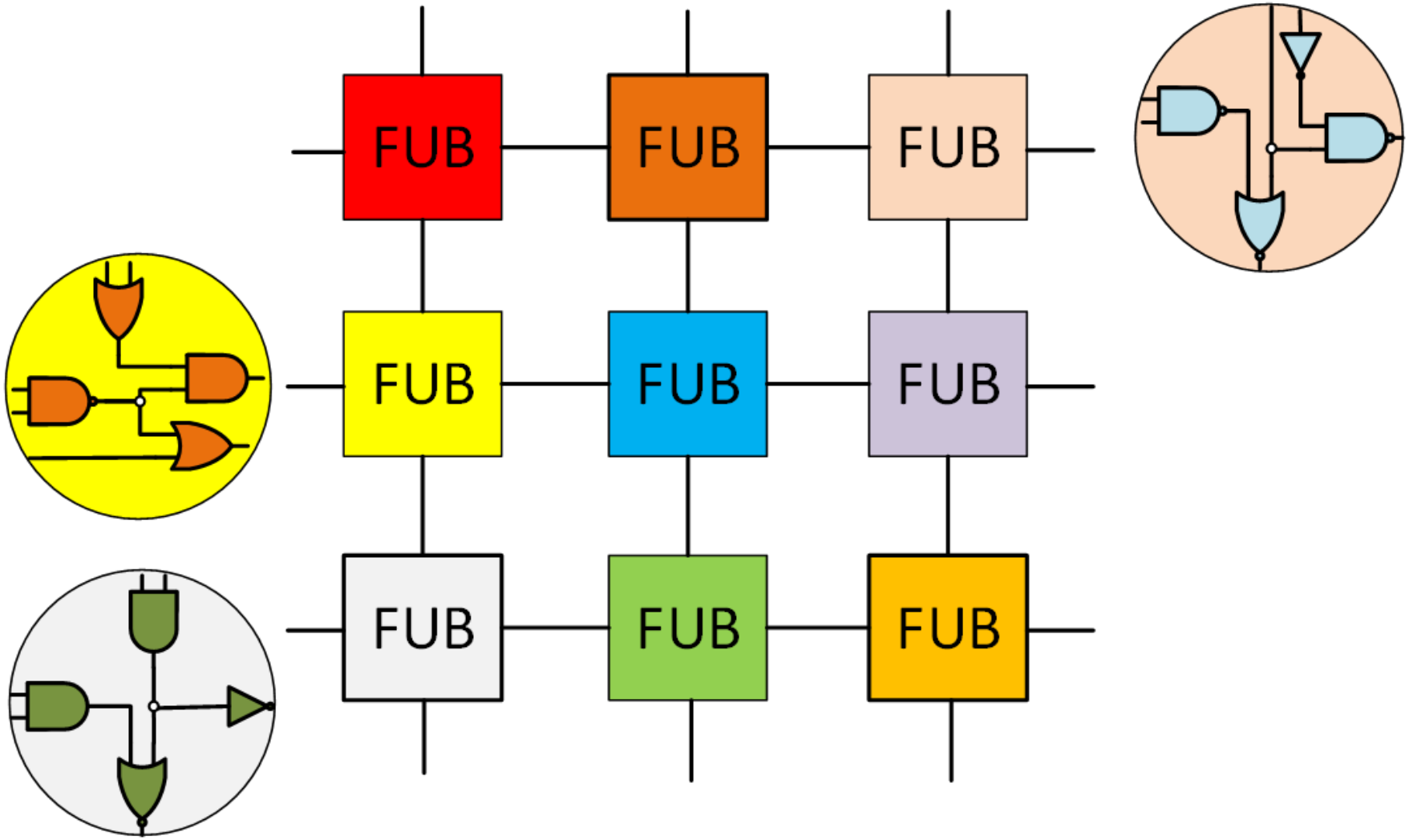
FUB Array Diagnosability

- Error pattern shown below is guaranteed
- **Top-Left** erroneous outputs identifies **defective FUB(s)**



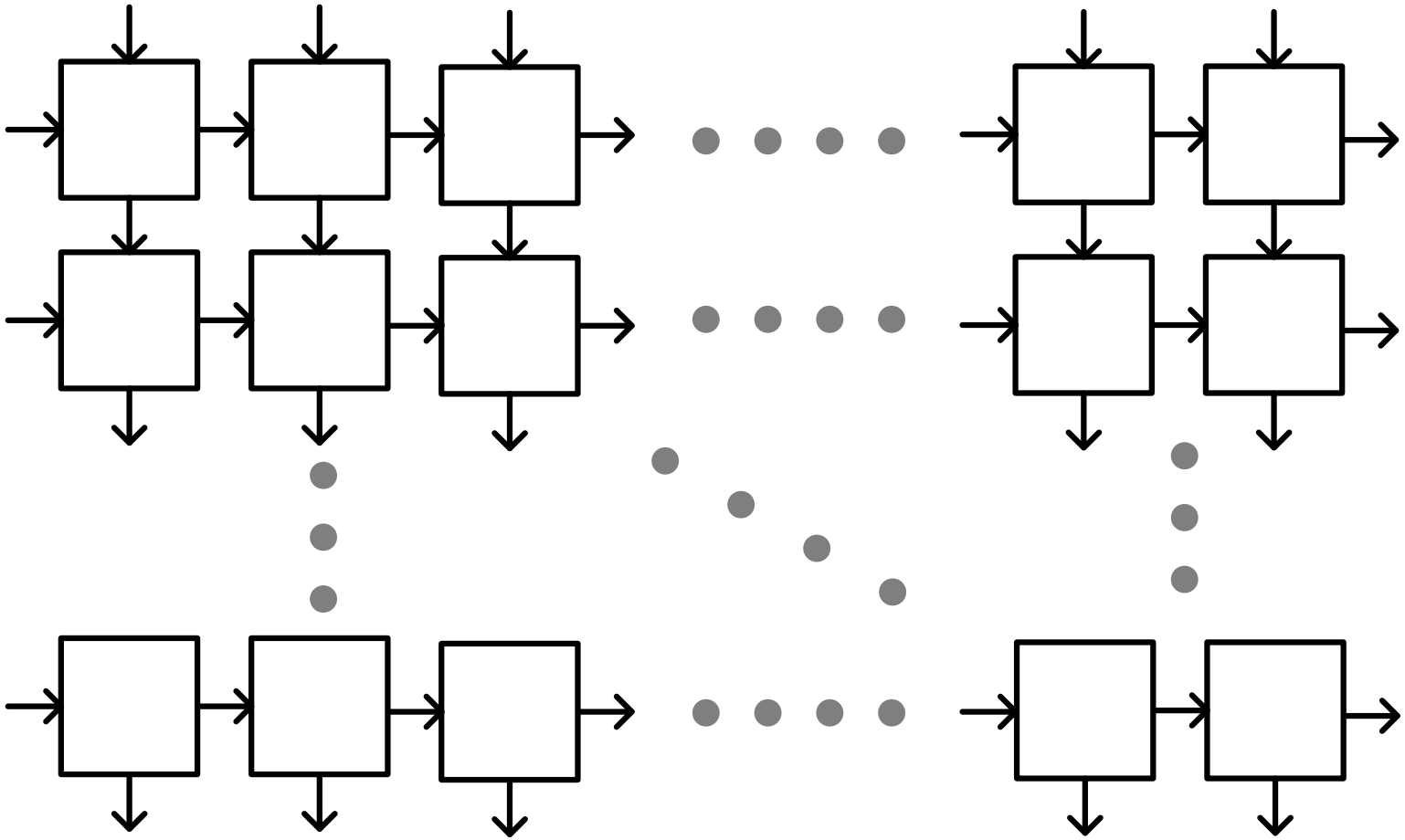
FUB Array Diversity

Each FUB function is bijective but logic can vary



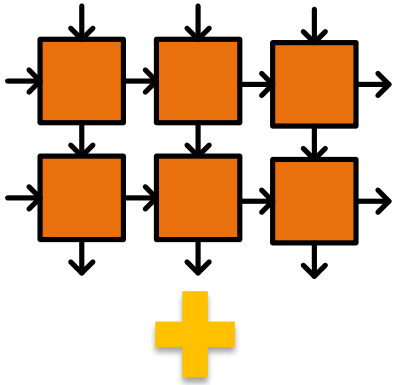
FUB Array Scalability

Testability and diagnosability are independent of size, shape



CM-LCV Test and Design Properties

Test and Diagnosis



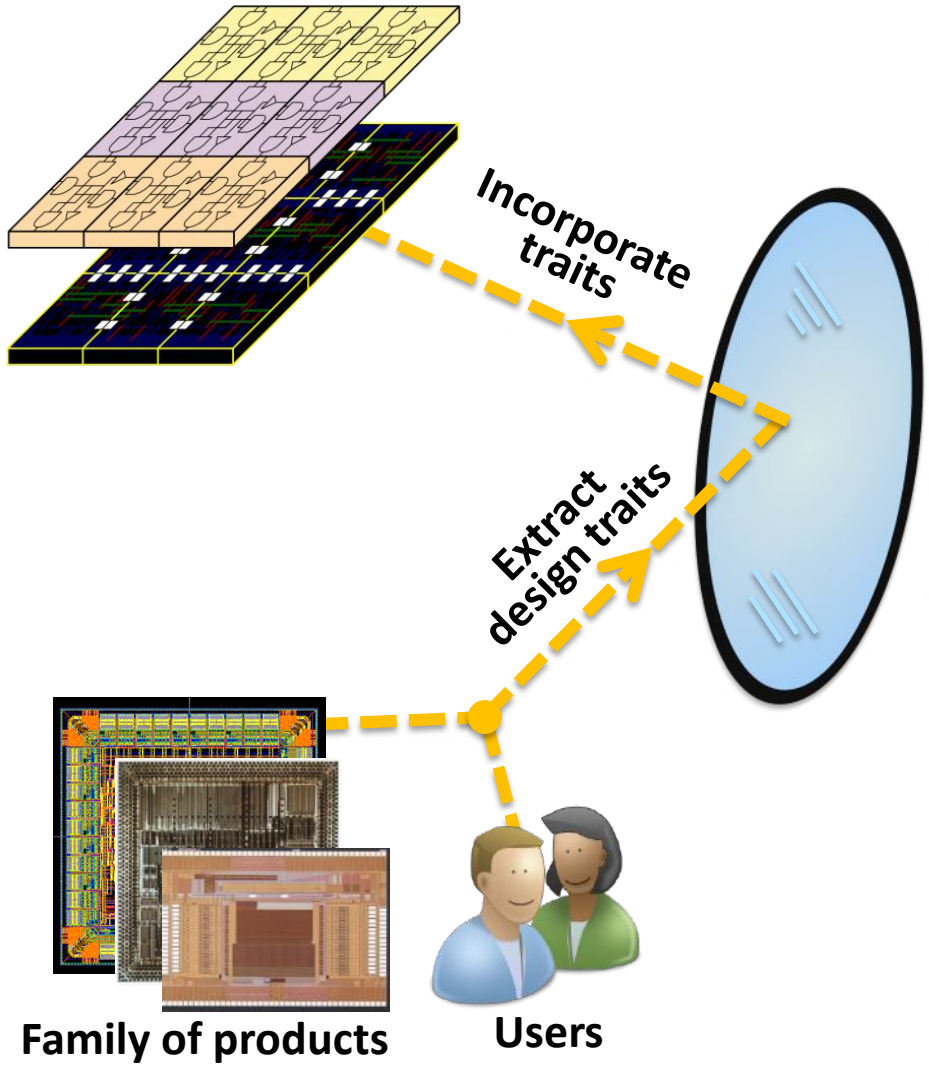
2D Array

IN	*OUT*
0000	1110
0001	1101
:	:
1111	1011

VH-Bijective FUB function

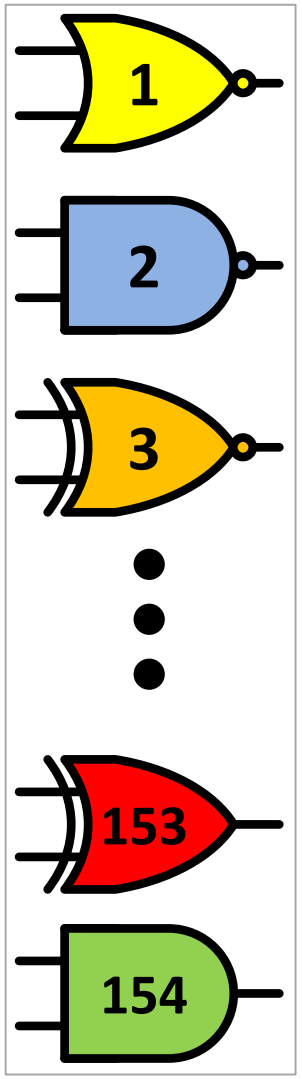
C-testability Efficient BIST
Ideal Diagnostic resolution
Ideal Diagnostic Accuracy
100% Pseudo-Exhaustive

Design Reflection

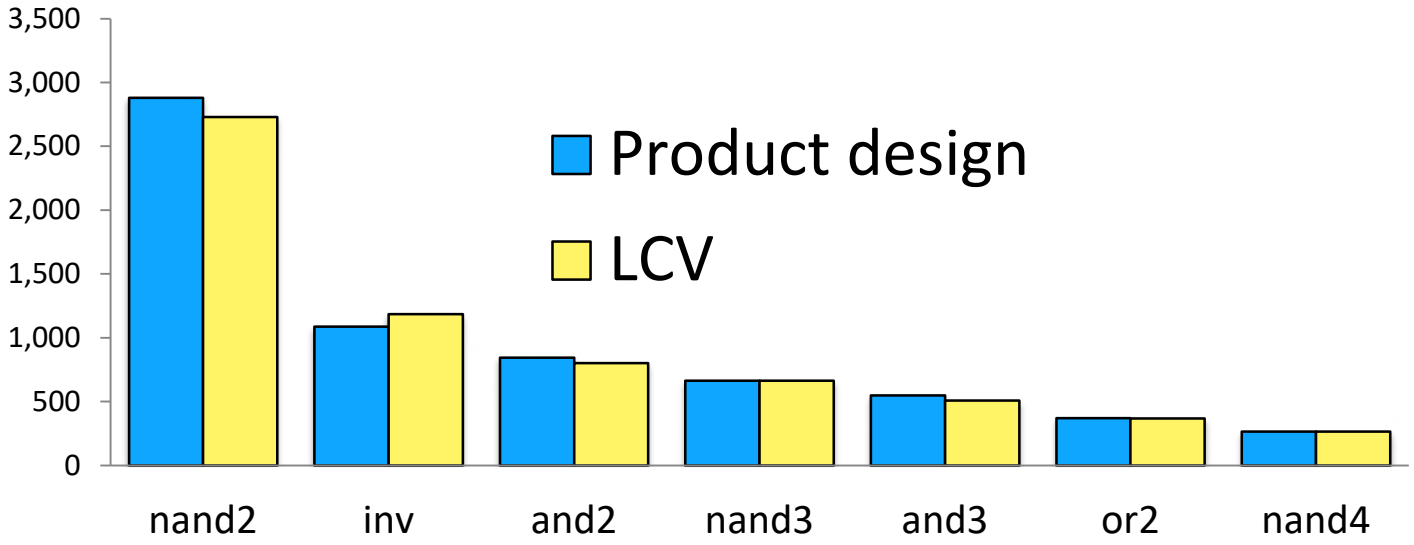


CM-LCV: Cell Library Charcterization

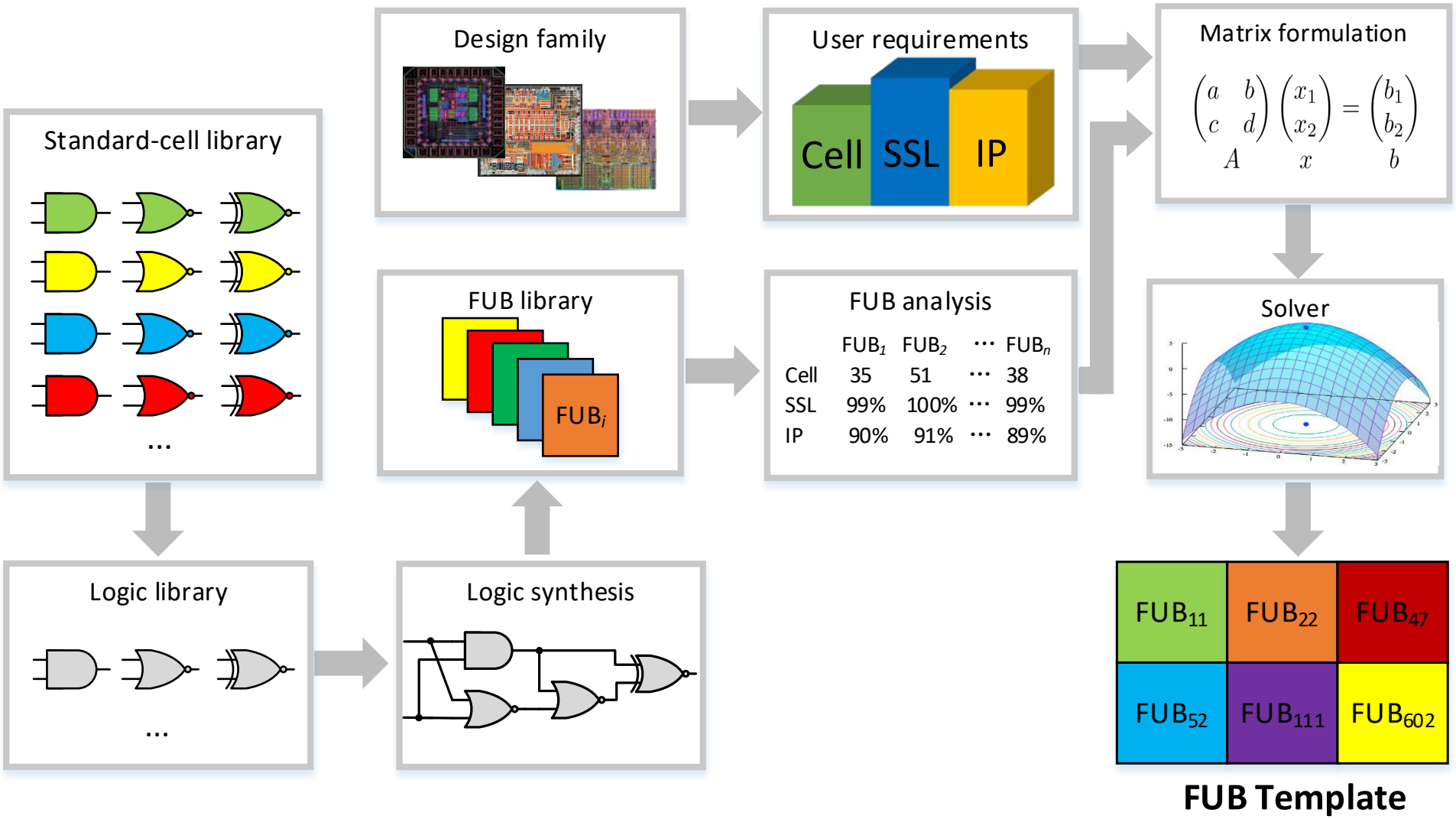
Cell library



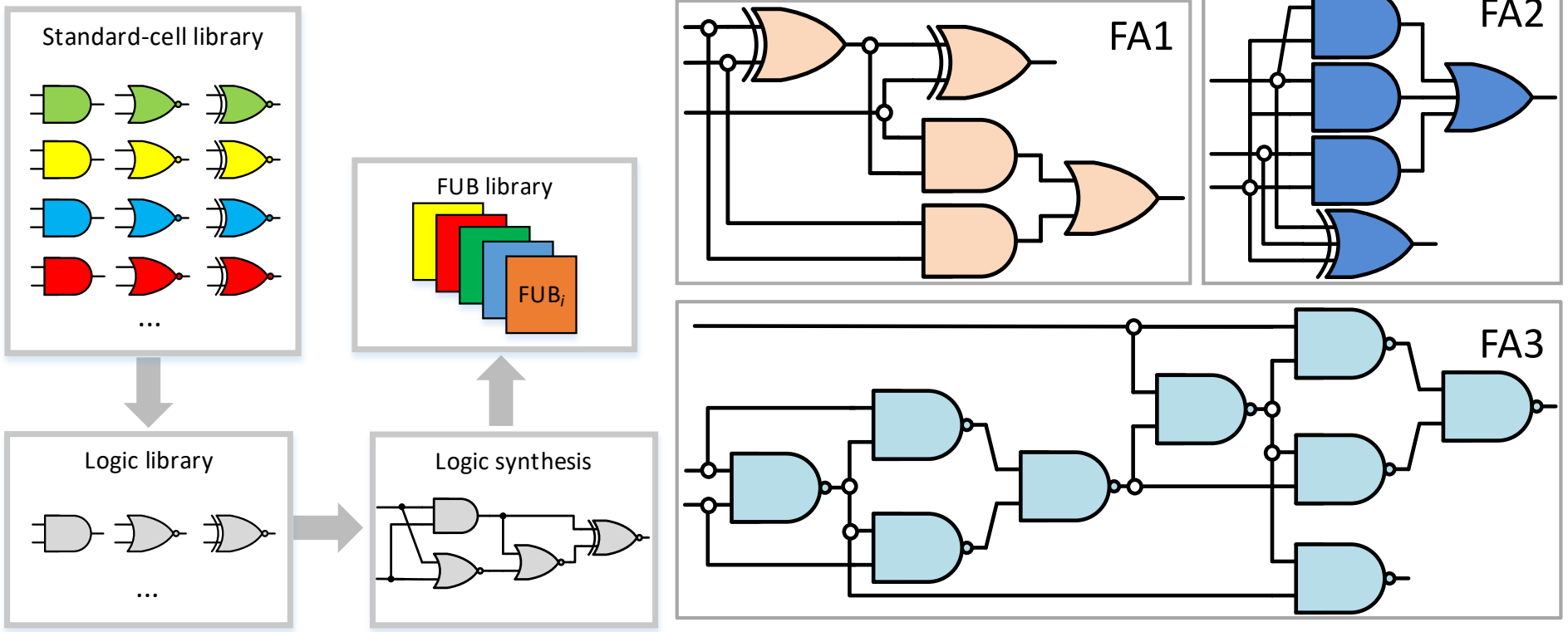
- Include all gates from the cell library
- Exhaustively test each cell (not just FUBs)
- Characterize cells that fail testing
- Make the LCV as small possible
- Make LCV design process automatable
- Make the LCV resemble real ICs



CM-LCV Design Flow

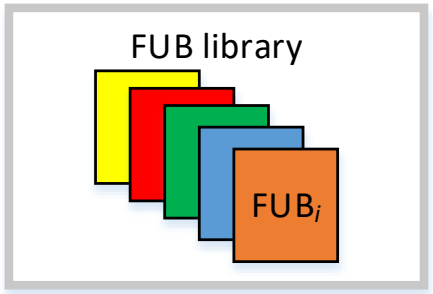


FUB Library Creation



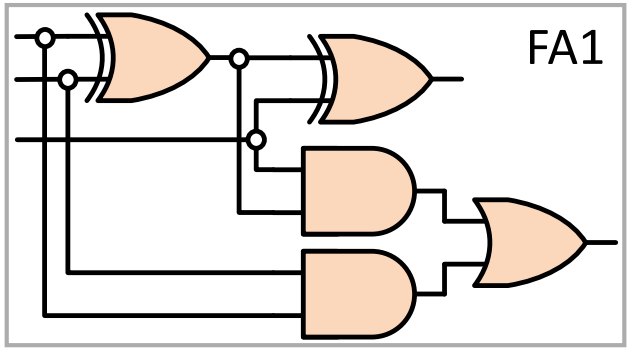
Circuit	NAND2	AND2	OR2	OR3	XOR2	XOR3
FA1	0	2	1	0	2	0
FA2	0	3	0	1	0	1
FA3	9	0	0	0	0	0

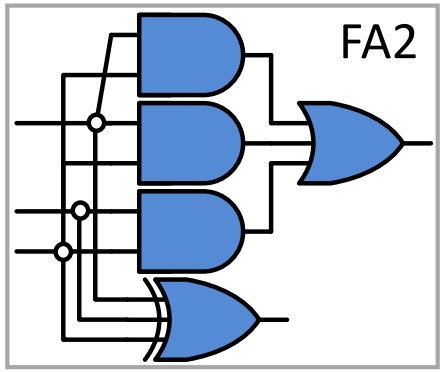
IP Fault Coverage Characterization

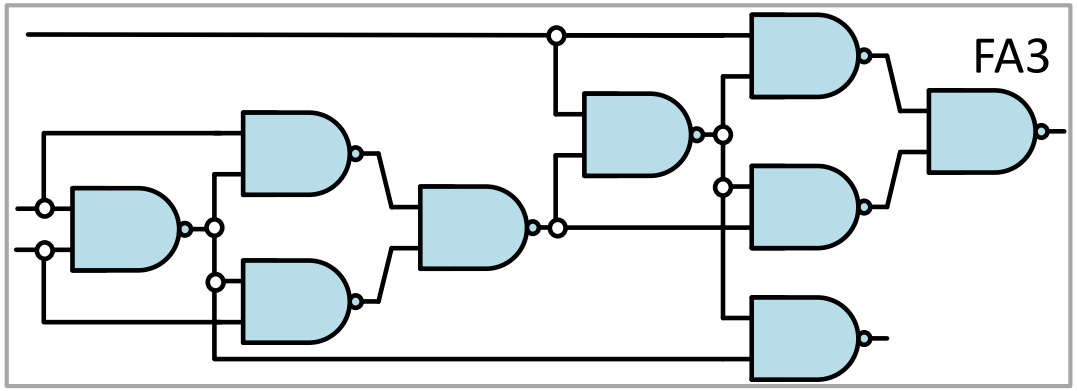


FUB analysis

	FUB_1	FUB_2	⋯	FUB_n
Cell	35	51	⋯	38
SSL	99%	100%	⋯	99%
IP	90%	91%	⋯	89%







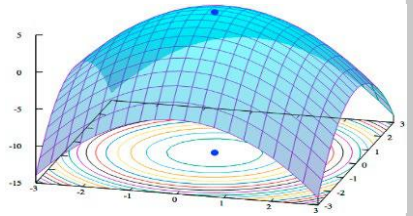
Circuit	IP fault coverage	Redundant instances	Redundant classes	Diagnostic coverage
FA1	95%	1	1	71.4%
FA2	89.2%	3	3	85%
FA3	80.6%	7	1	58.3%

FUB Template Formation

Matrix formulation

$$\underset{A}{\begin{pmatrix} a & b \\ c & d \end{pmatrix}} \underset{x}{\begin{pmatrix} x_1 \\ x_2 \end{pmatrix}} = \underset{b}{\begin{pmatrix} b_1 \\ b_2 \end{pmatrix}}$$

Solver



FUB_{11}	FUB_{22}	FUB_{47}
FUB_{52}	FUB_{111}	FUB_{602}

Design reflection

$$\begin{bmatrix} 0 & 0 & 9 \\ 2 & 3 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 2 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \cdot \begin{bmatrix} FA1 \\ FA2 \\ FA3 \end{bmatrix} \cong \begin{bmatrix} 900 \\ 360 \\ 150 \\ 30 \\ 805 \\ 200 \end{bmatrix}$$

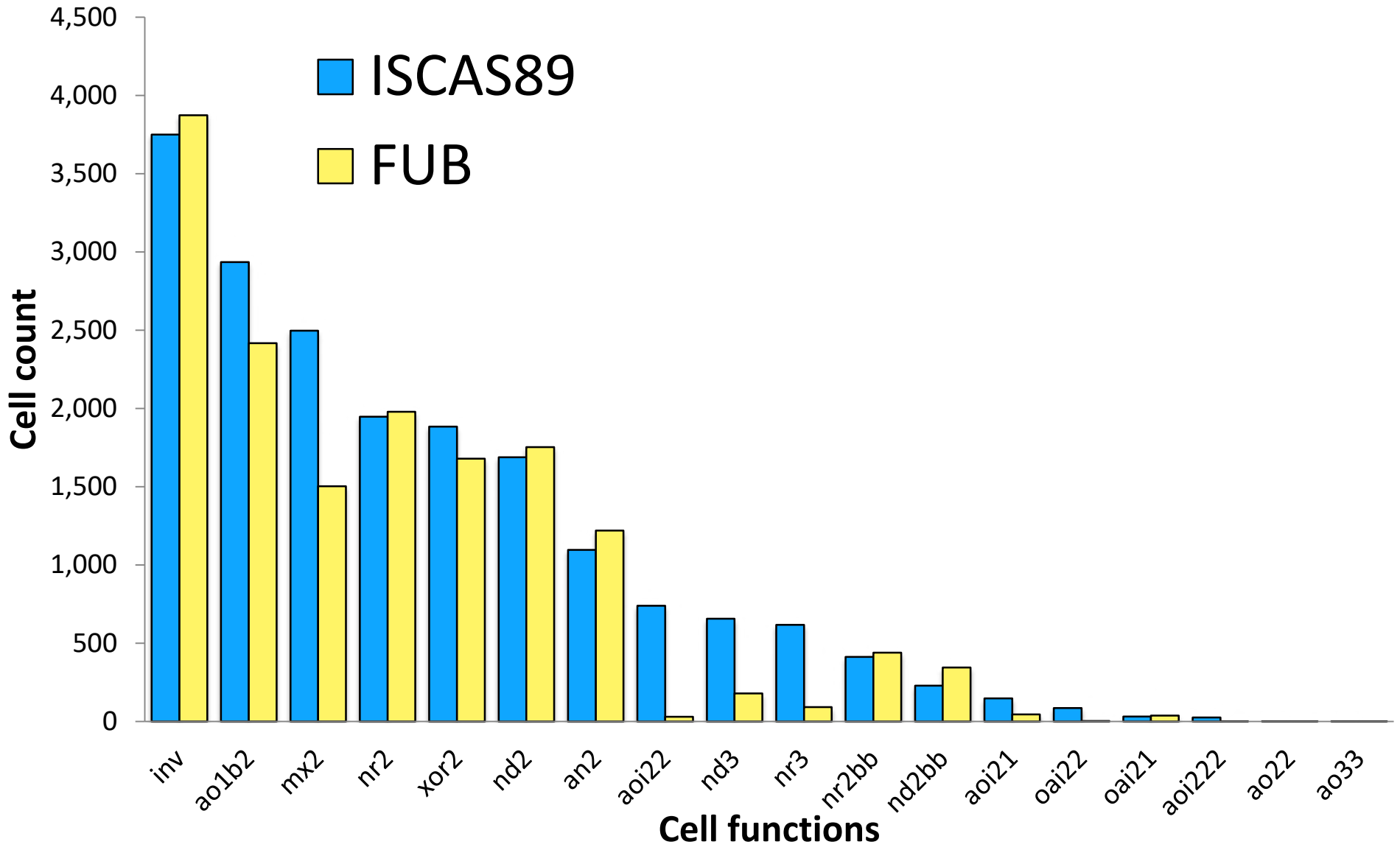
IP fault instance detection

$$\begin{bmatrix} 1 & 3 & 7 \end{bmatrix} \cdot \begin{bmatrix} FA1 \\ FA2 \\ FA3 \end{bmatrix} \cong \begin{bmatrix} 0 \end{bmatrix}$$

IP fault class detection

$$\begin{bmatrix} 0 & 0 & 1 \\ 1 & 0 & 1 \\ \vdots & \vdots & \vdots \end{bmatrix} \cdot \begin{bmatrix} FA1 \\ FA2 \\ FA3 \end{bmatrix} \geq \begin{bmatrix} 1 \\ 1 \\ \vdots \end{bmatrix}$$

ISCAS89 Design Reflection

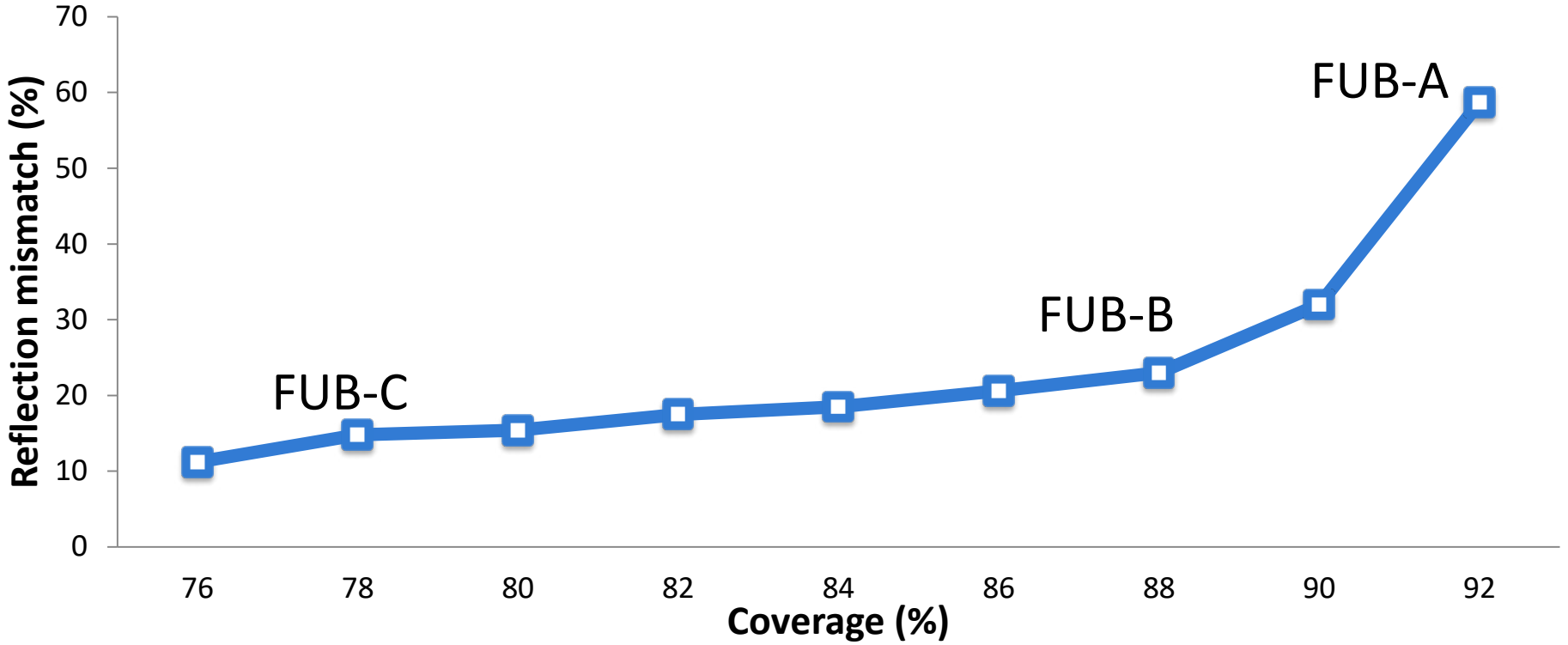


Fault Coverage

Circuit	SSL fault		IP fault		IP fault classes	
	Coverage	No. of tests	Coverage	No. of tests	Missing	Redundant
ISCAS89	**99.5%**	584	89.4%	1,336	1,372	2
ITC99	99.0%	5,026	72.0%	10,583	1,186	6
FUB	99.4%	**64**	**92.4%**	**64**	**0**	**0**

Circuit	Inputs	Outputs	Cell counts
ISCAS89	6,202	6,994	18,741
ITC99	13,239	13,280	190,394
FUB (6 I/O, 12x12)	**72**	**72**	**7,552**

ISCAS89 Reflection-Testability Tradeoff



Circuit	FUB I/O	FUB template	Cell counts	FUB array size
FUB-A	6	139	7,552	12x12
FUB-B	6	287	15,594	17x17
FUB-C	6	319	17,696	18x18

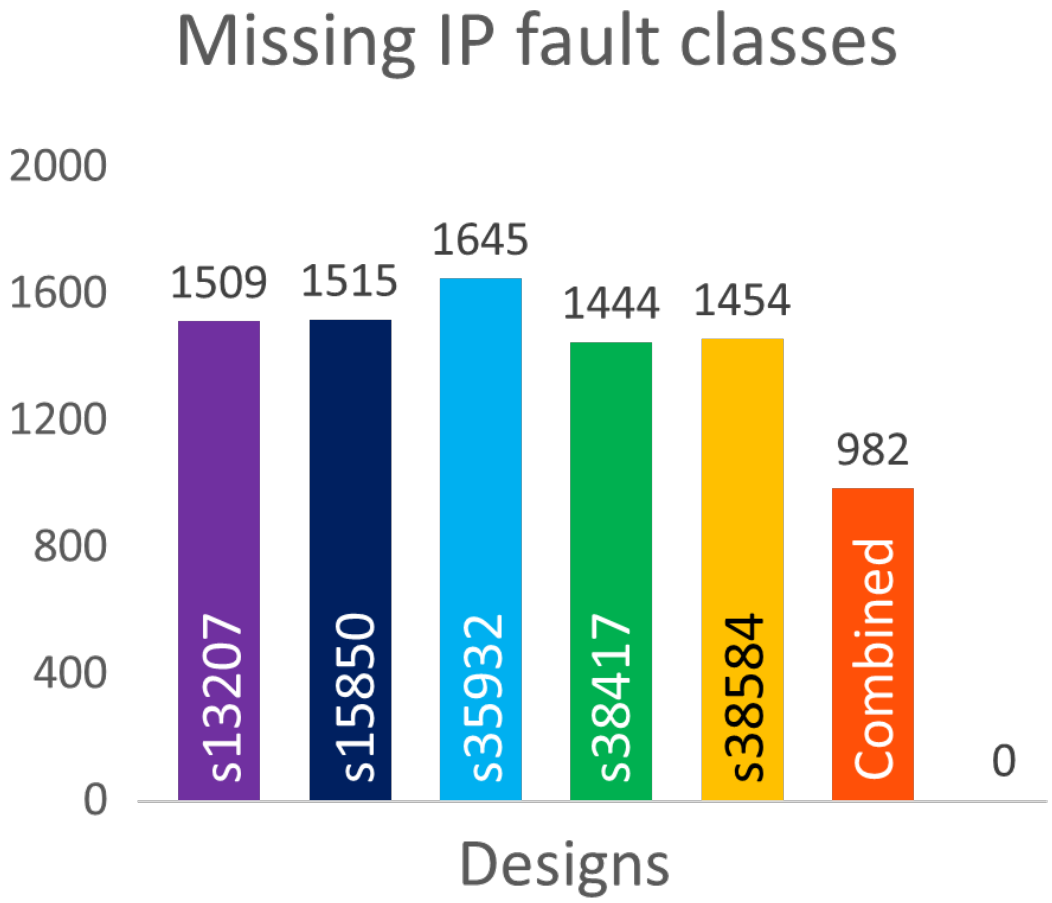
Defect Coverage

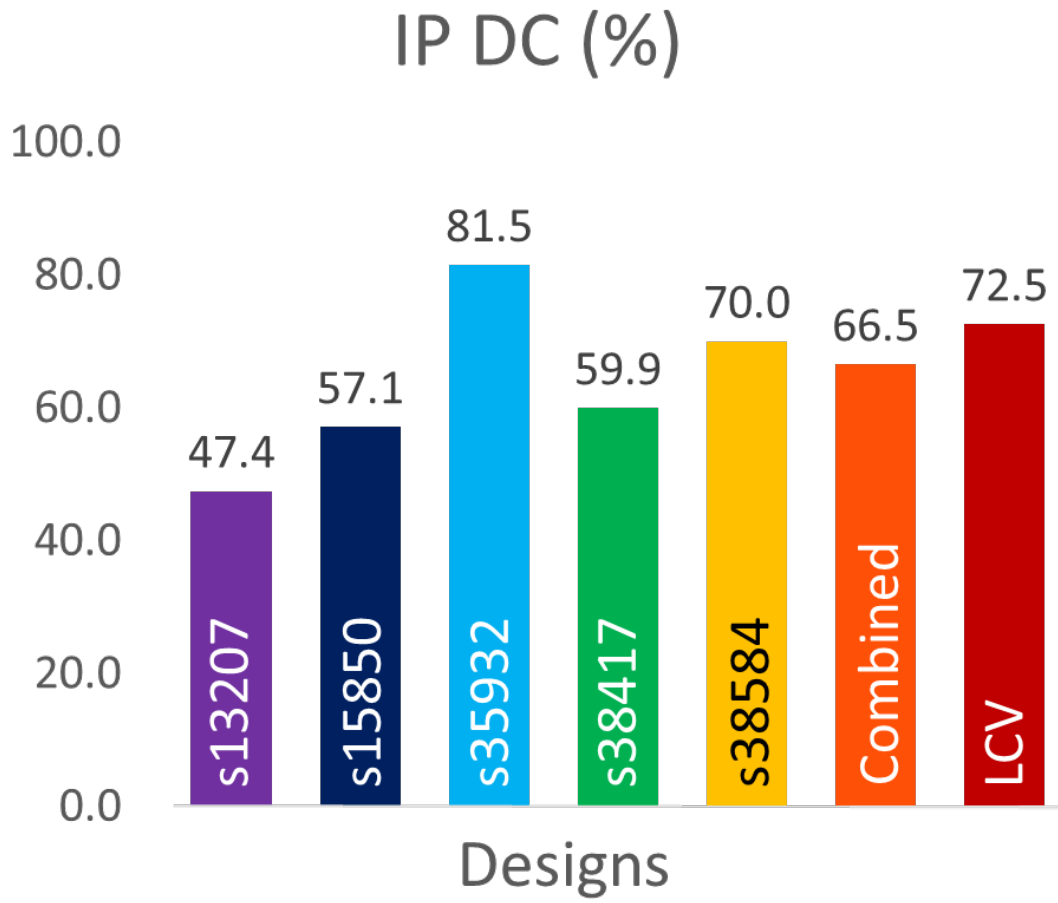
SLIDER Tool [VTS'11, TCAD'12]

- **Layout-level defect-induction**
- **Circuit-level defect simulation**
- **Applied to all cells (including x1,x2, etc.)**

Circuit	Redundant defects	Defect coverage
ISCAS89	2,079	96.6%
ITC99	54,677	90.0%
FUB Array	**0**	**100%**

Diagnostic Coverage



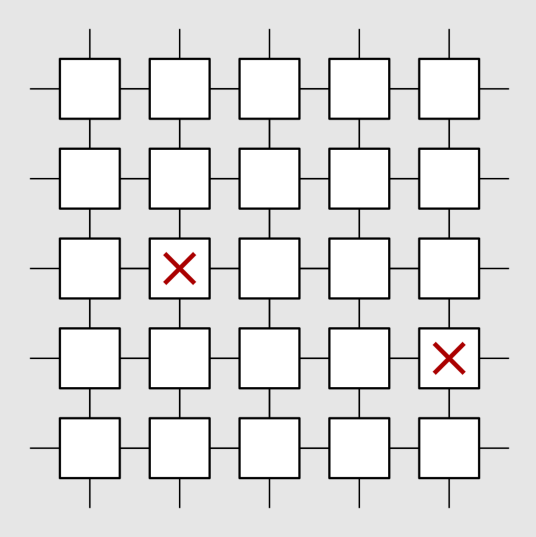


IP fault class = truth table entry of a standard cell

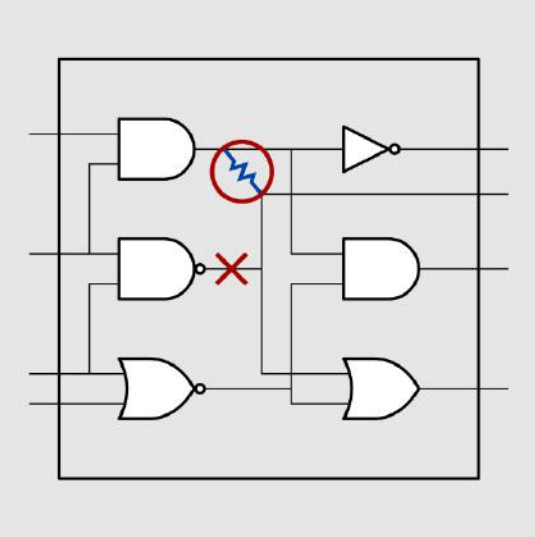
$$DC = \frac{No.\ of\ fault\ signatures}{Faults}$$

Multi-Level Diagnosis

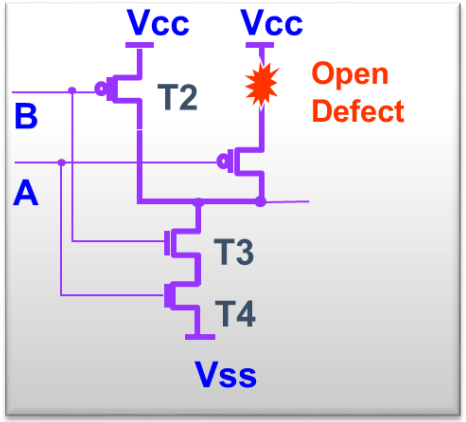
Array level



FUB level



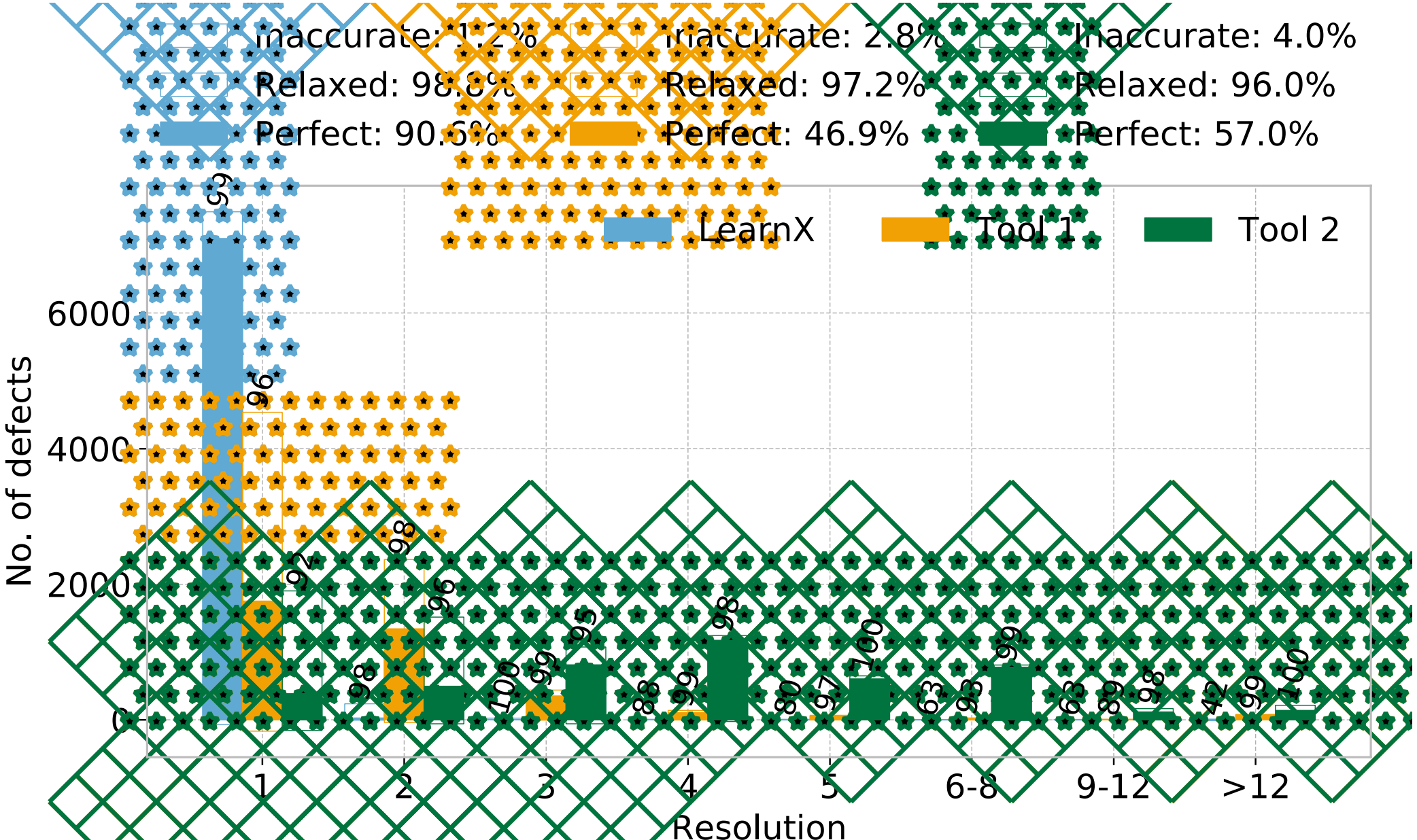
Cell level



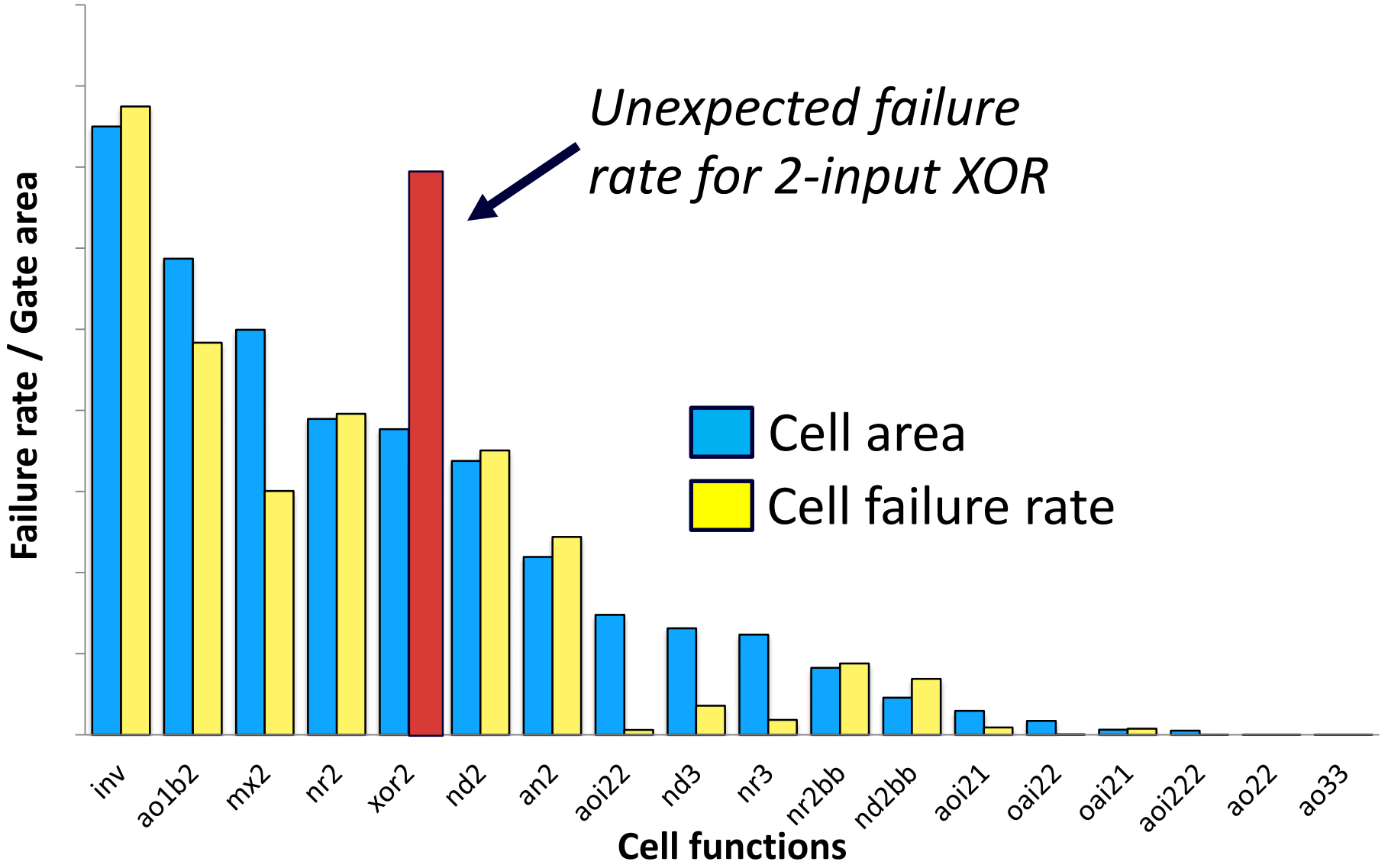
1. **Array: Identify defective FUBs (custom tool)**
2. **FUB: Identify defective interconnects**
3. **Cell: Identify defective cells**

Multiple defect diagnosis feasible because FUB size < few hundred gates

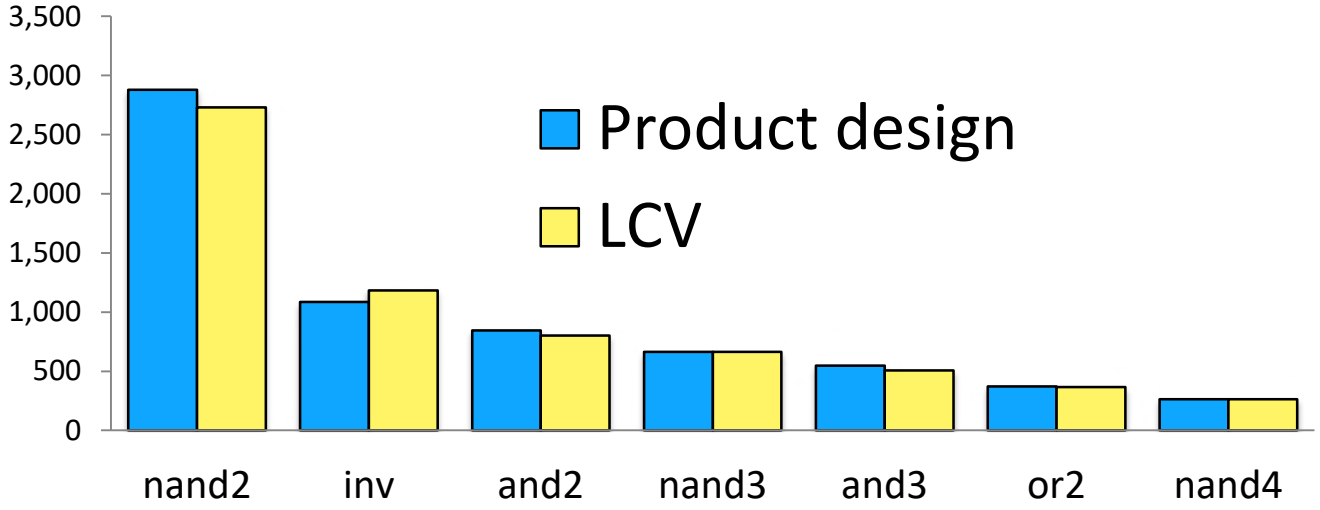
Diagnostic Data Analytics

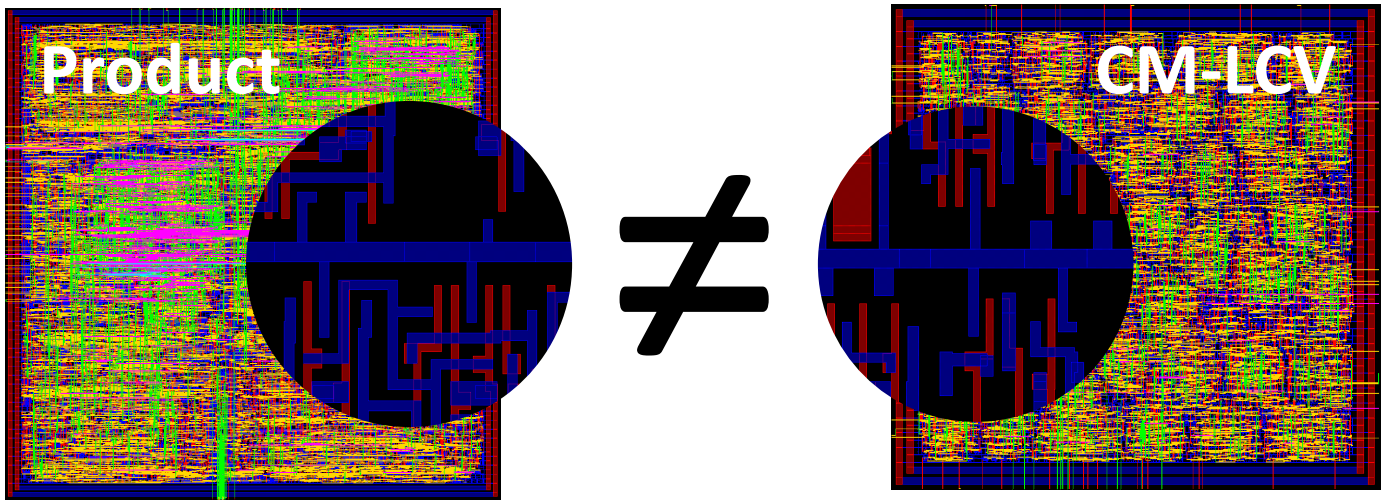


Outcome of LCV Fabrication



Logic .vs. Layout Reflection



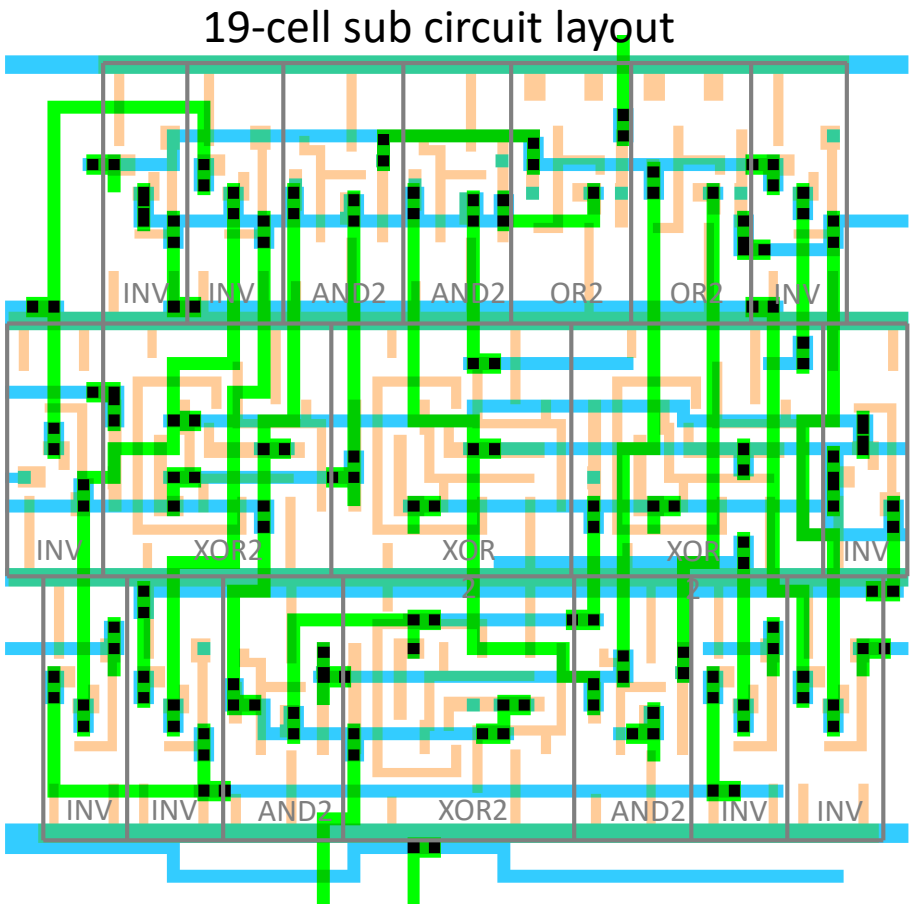


With nearly identical usage of standard cells, fewer than 0.001% common layout patterns

Reflection

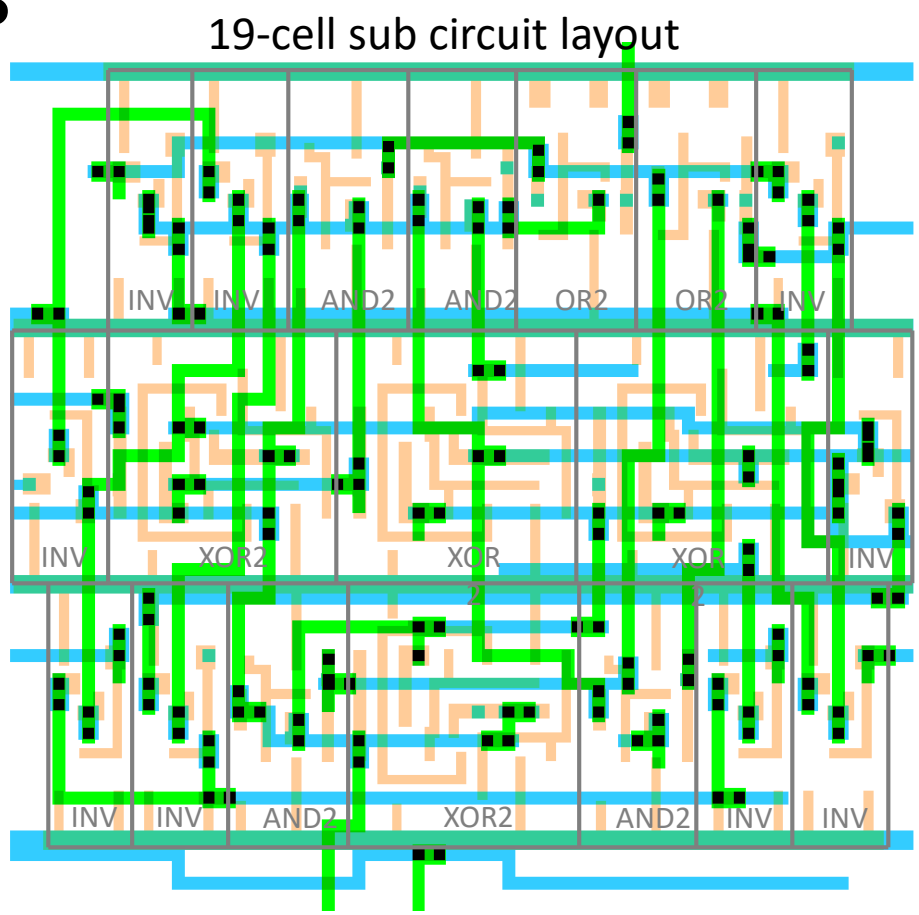
Layout demographics are separated into 3 categories:

- **Intra-cell**
- **Inter-cell**
- **Inter-connect**



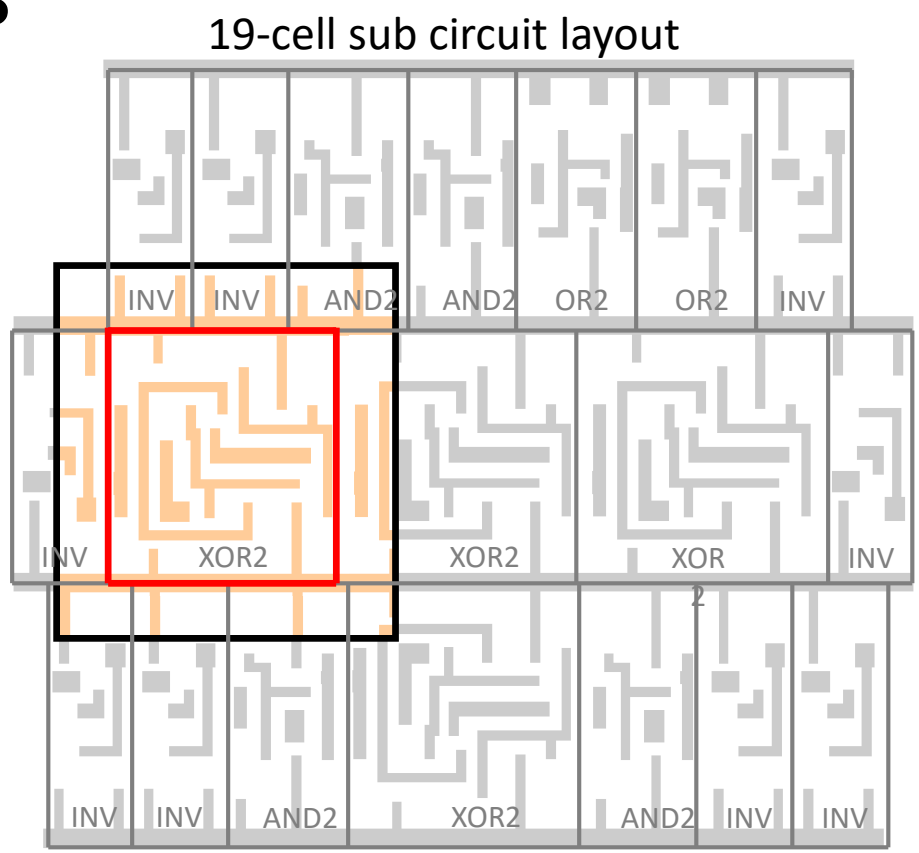
Inter-cell Reflection

- **Focus on cell neighborhoods**
- **Goal: match cell neighborhoods**



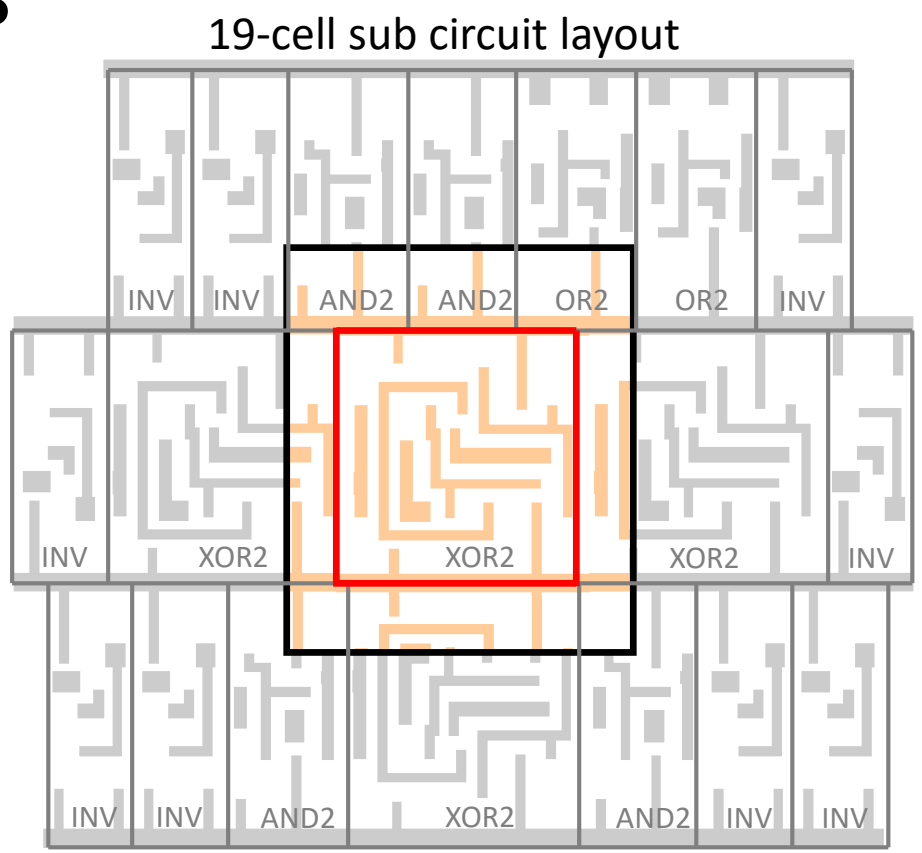
Inter-cell Reflection

- **Focus on cell neighborhoods**
- **Goal: match cell neighborhoods**



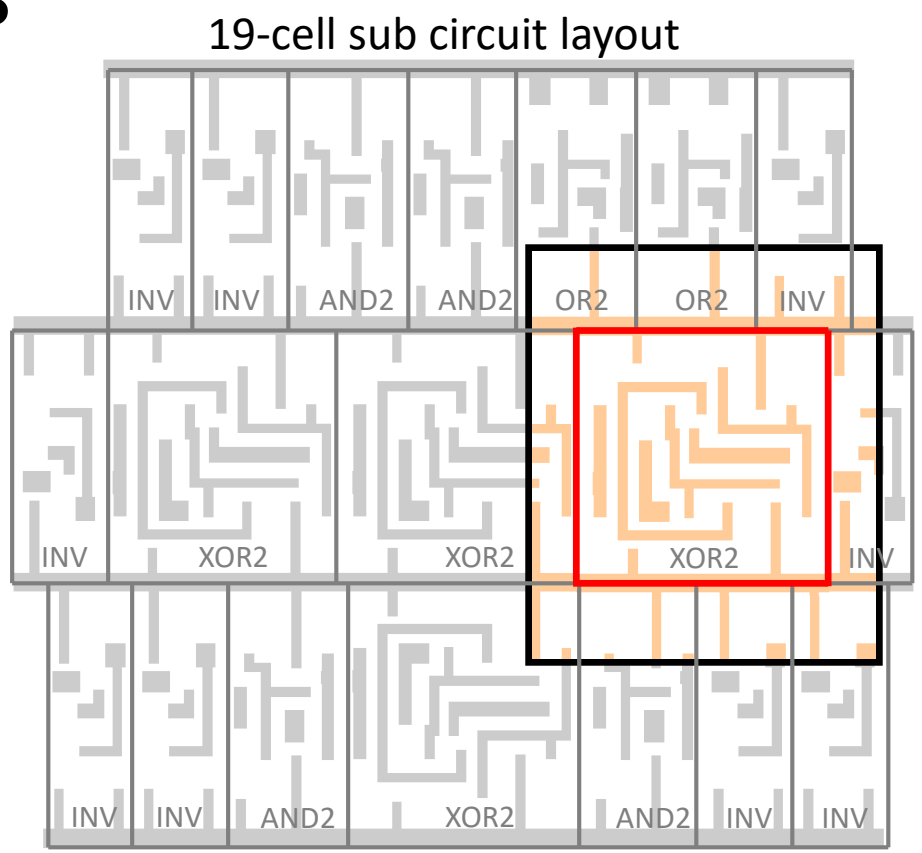
Inter-cell Reflection

- **Focus on cell neighborhoods**
- **Goal: match cell neighborhoods**



Inter-cell Reflection

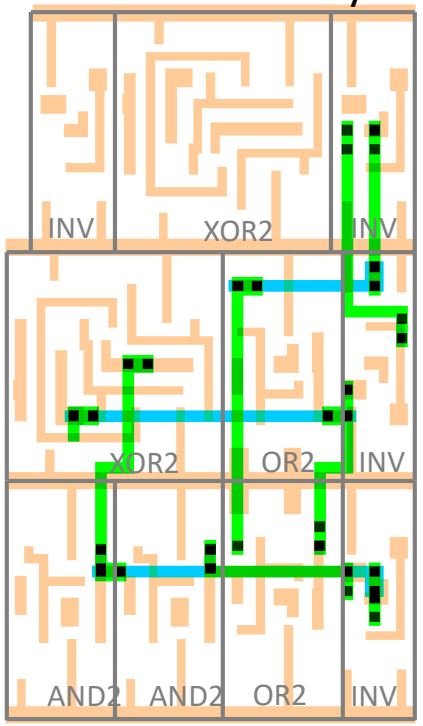
- **Focus on cell neighborhoods**
- **Goal: match cell neighborhoods**



Inter-cell Reflection

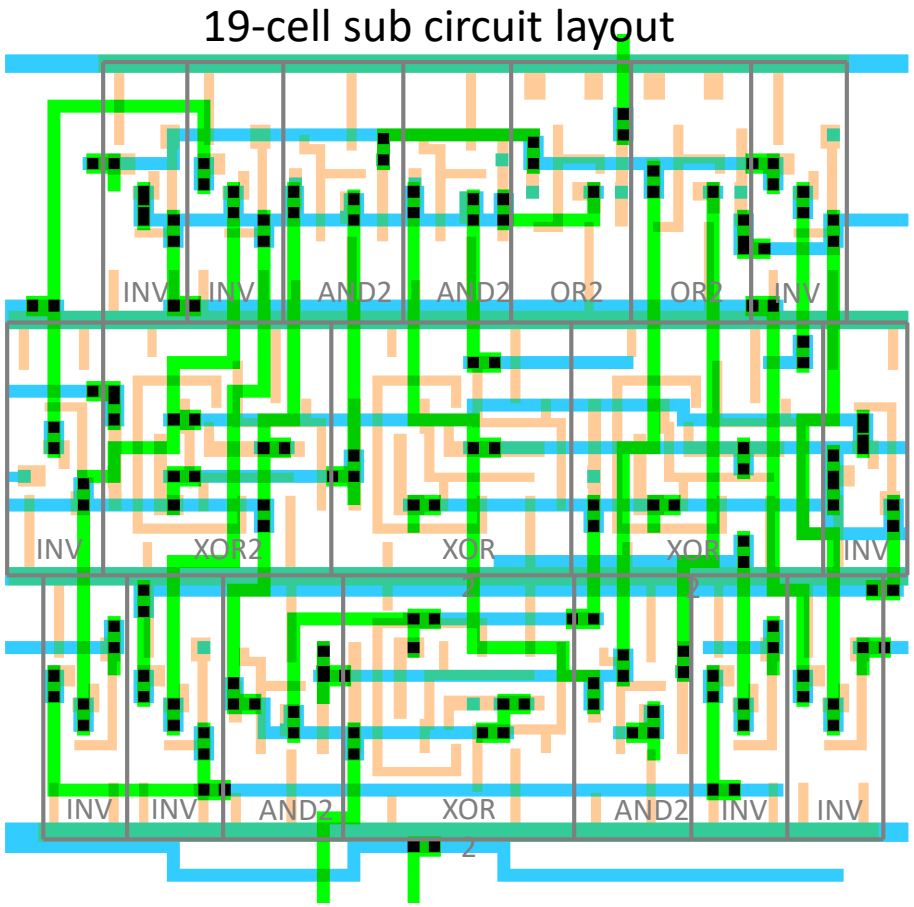
- Layout rewiring [ITC'16]
  - Select a representative product design
  - Leave placed cells, remove all wires
  - Rewire cells to create CM-LCV functionality
- Inter-cell layout geometries incorporation [ITC'17]
  - Identify the representative neighborhoods
  - Create CM-LCV functionality with all cell neighborhoods of interest
  - Incorporate inter-cell layout via place and route

10-cell sub circuit layout



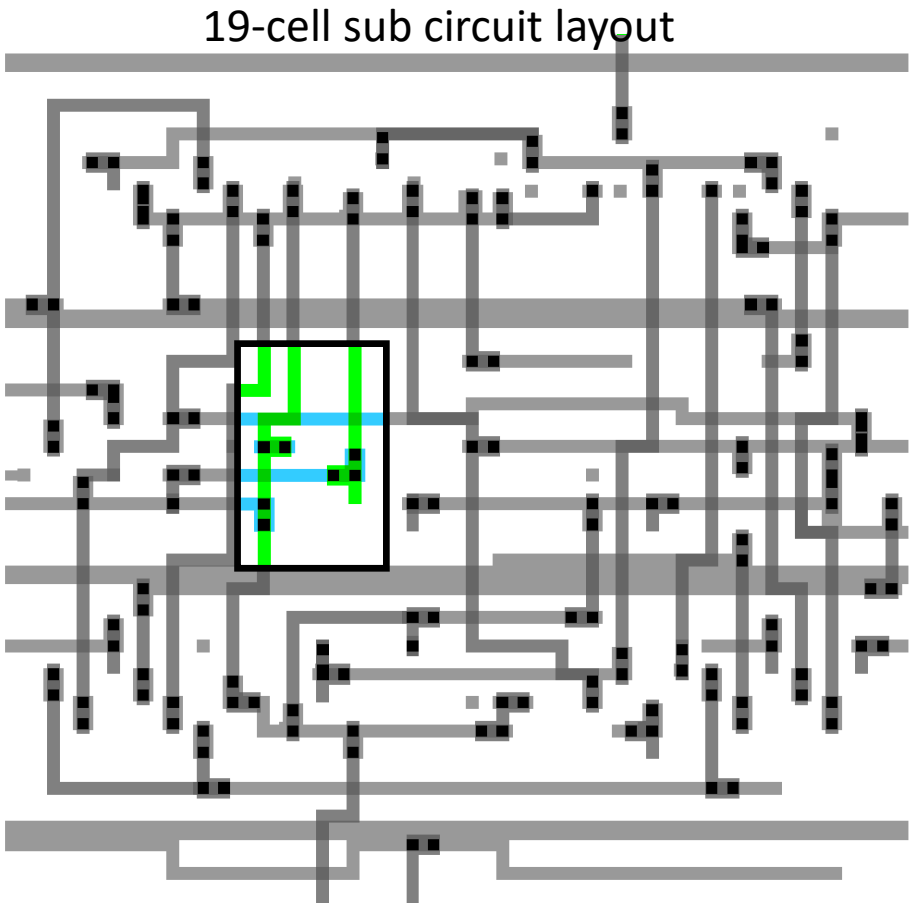
Inter-connect Reflection

- **Focus on inter-connect**
- **Inter-connect = wires, vias**
- **Goal: match interconnect geometries**



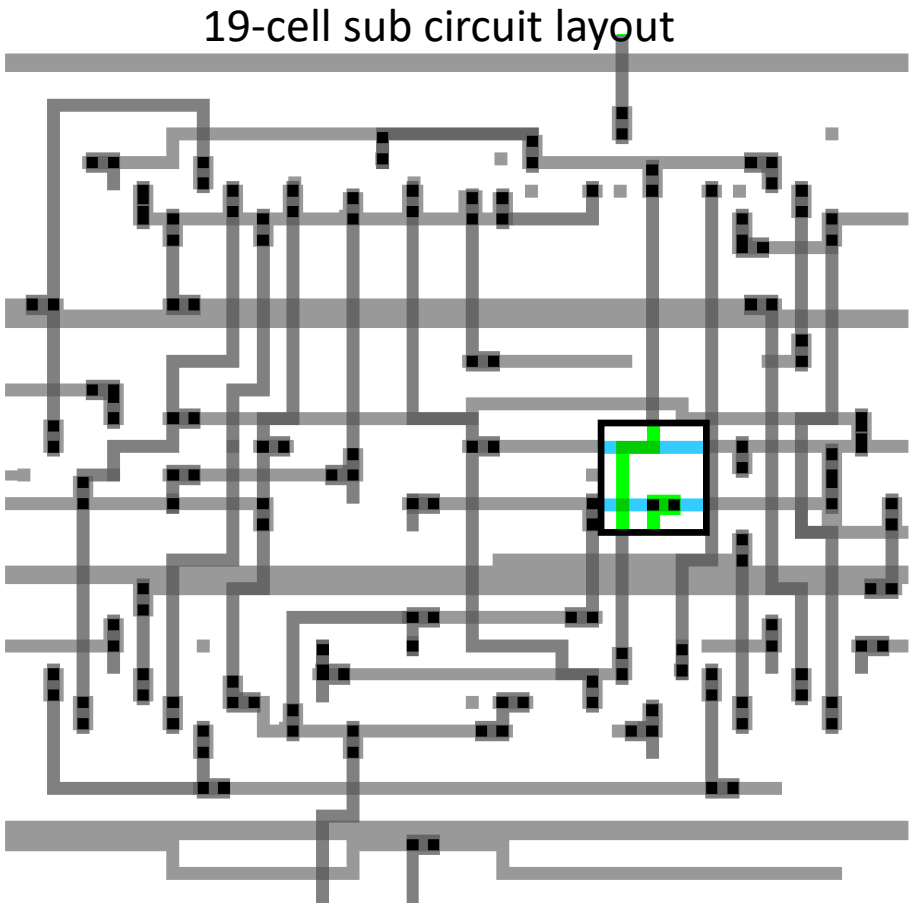
Inter-connect Reflection

- **Focus on inter-connect**
- **Inter-connect = wires, vias**
- **Goal: match interconnect geometries**



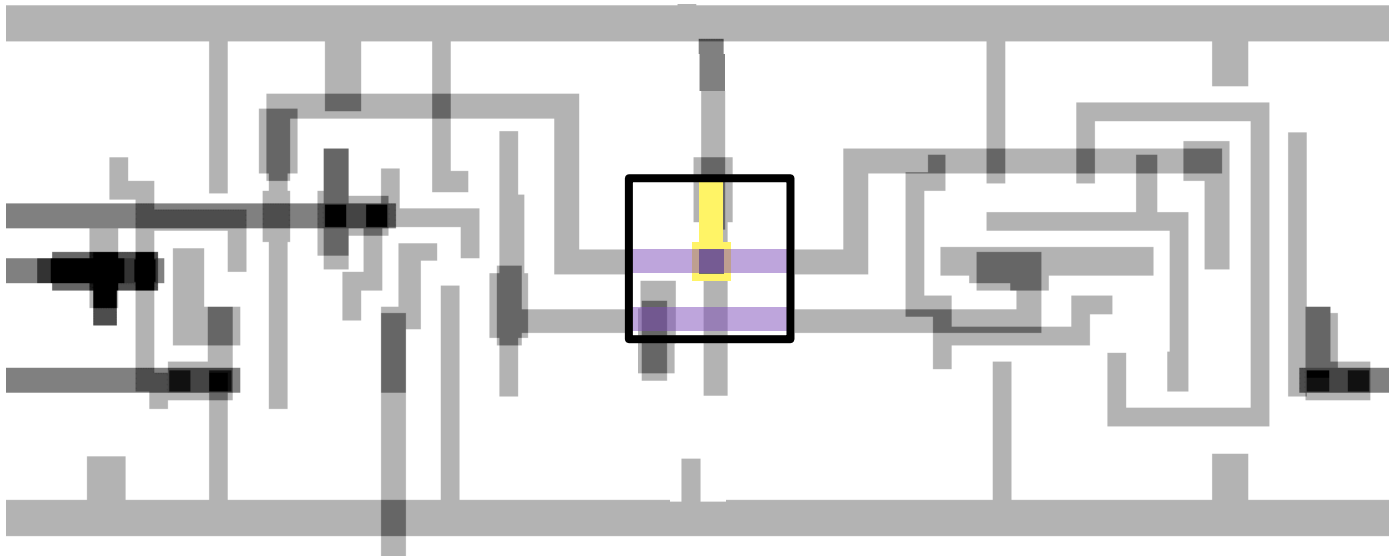
Inter-connect Reflection

- **Focus on inter-connect**
- **Inter-connect = wires, vias**
- **Goal: match interconnect geometries**

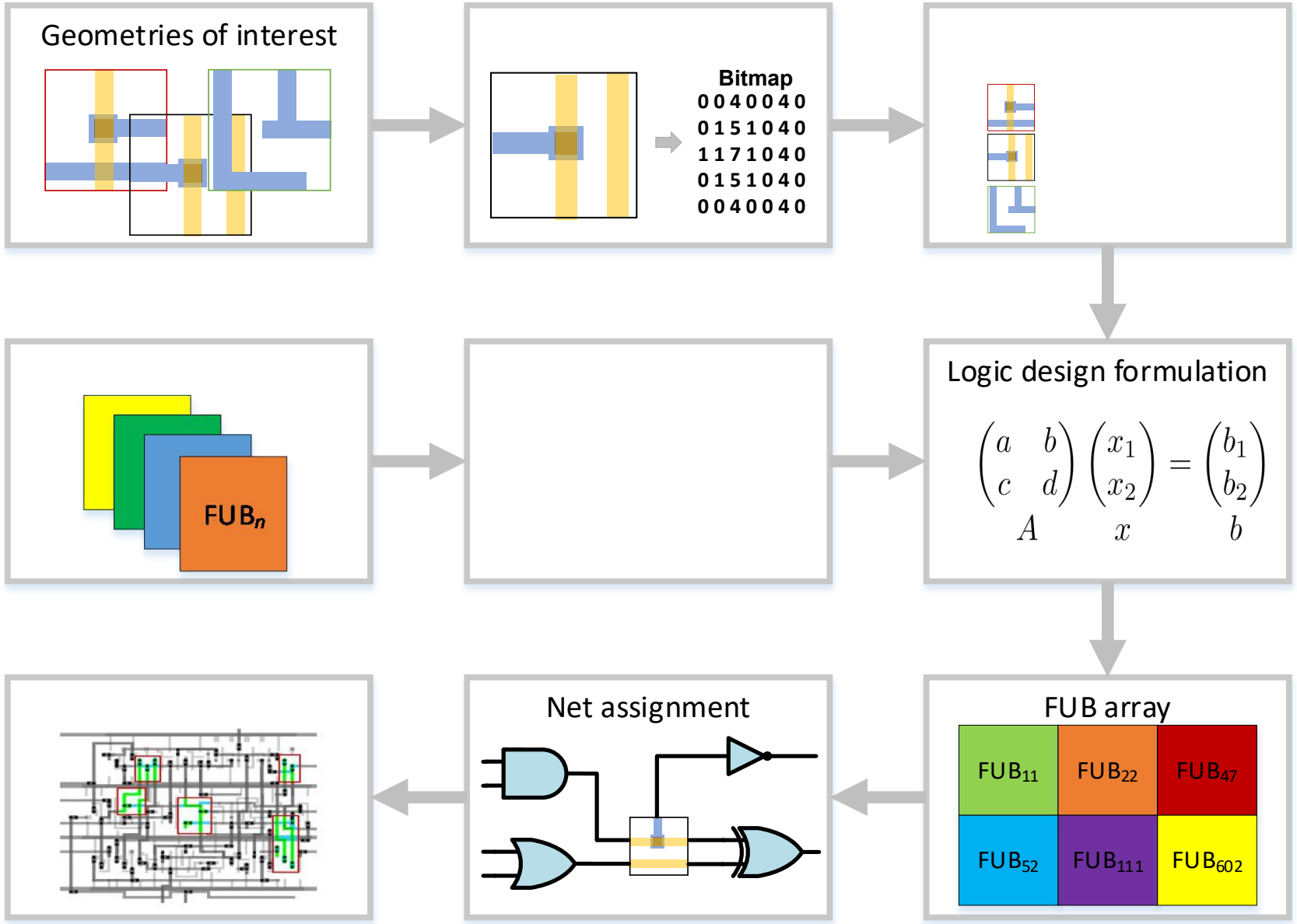


Key Insight: Inter-connect Reflection

- Treat the layout geometries of interest as logic components
- Identify the logic for “component” incorporation
- Optimize routing and placement for physical implementation

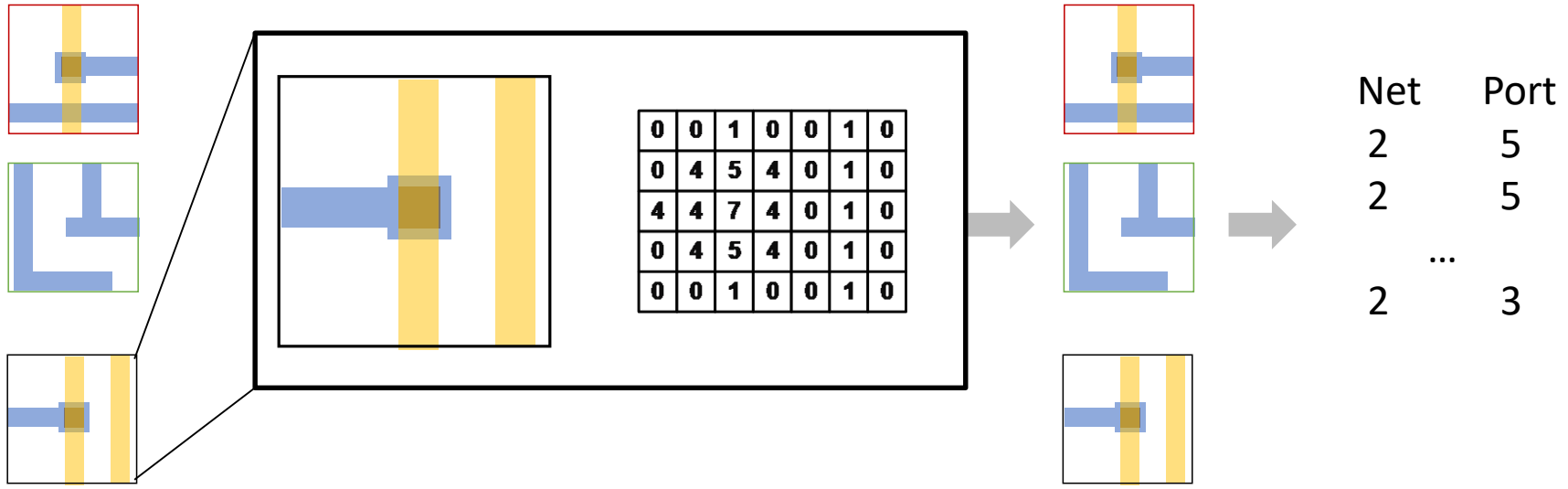


Design Flow for Inter-connect Reflection



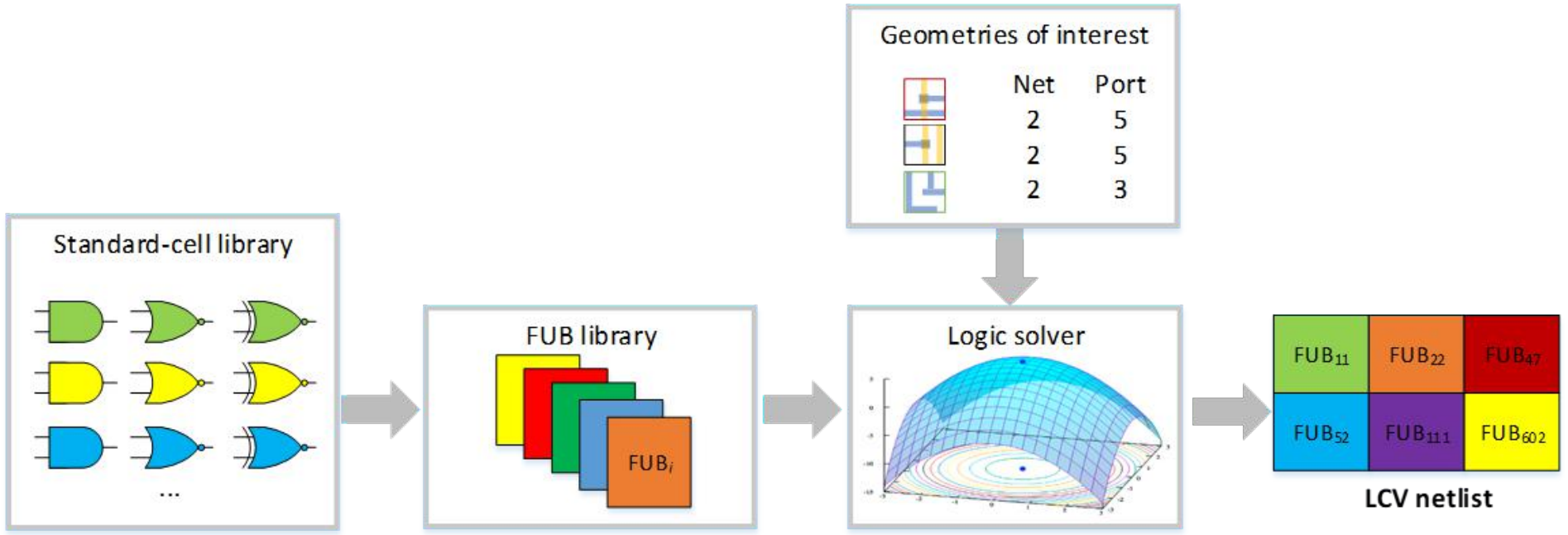
Layout Geometry Characterization

- Identify layout geometries of interest
- Describe layout geometries as topological patterns
- Abstract layout geometries to the logical level



LCV Netlist Generation

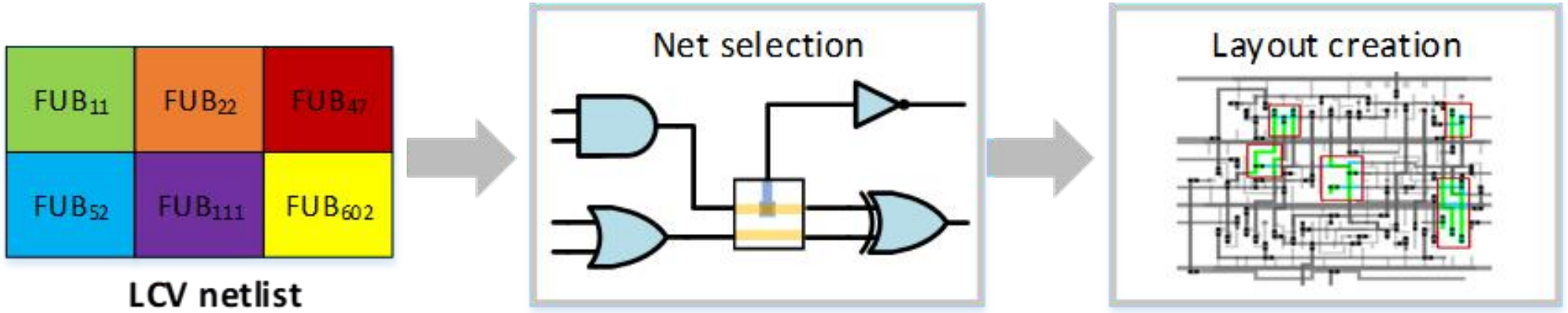
- Create and characterize a FUB library for layout geometries
- Add optimization constraints for layout inclusion
- Optimization creates array with cells for layout geometries



41

Layout Geometries Physical Implementation

- Layout geometries are not incorporated automatically
- Optimization creates net connection to layout geometries
- Creating LCV layout via placement and routing



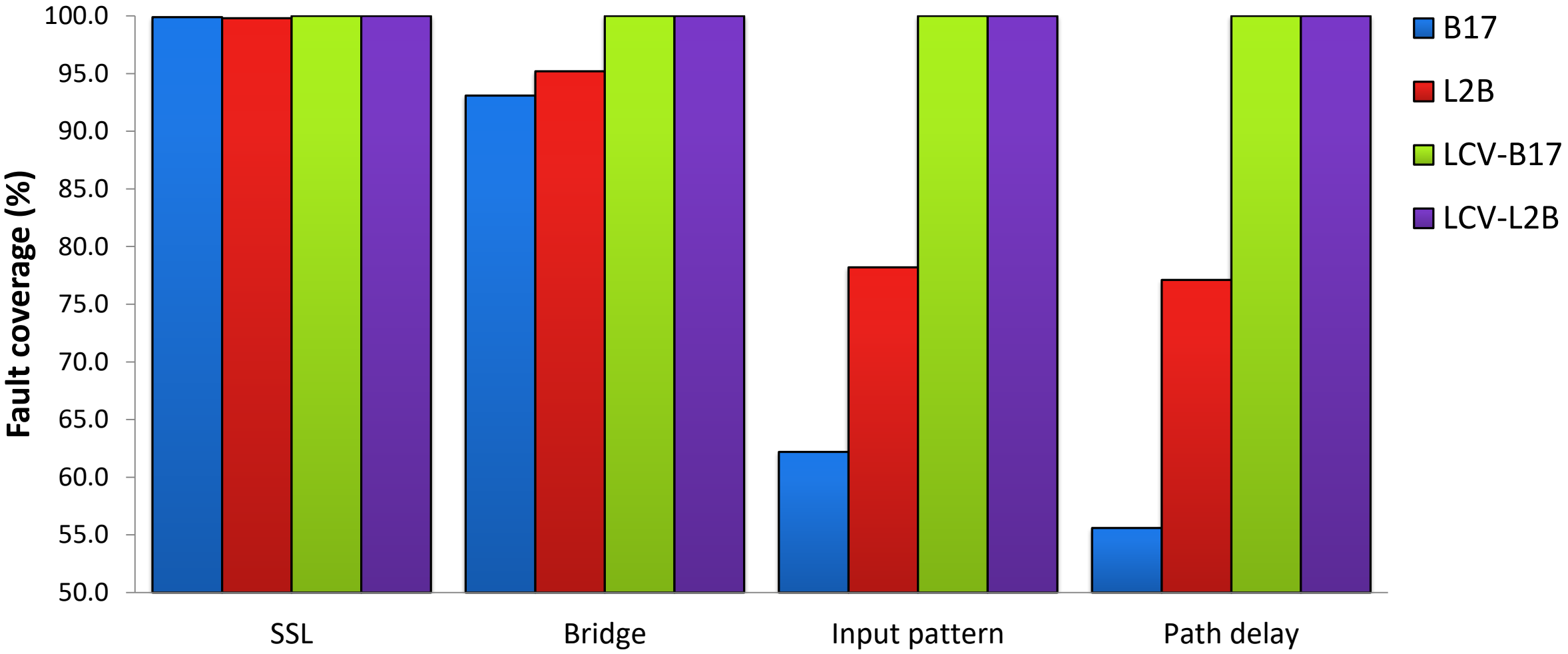
Design Experiments

- **Objective**: Evaluate LCV inter-connect reflection design flow
  - Defect testability:

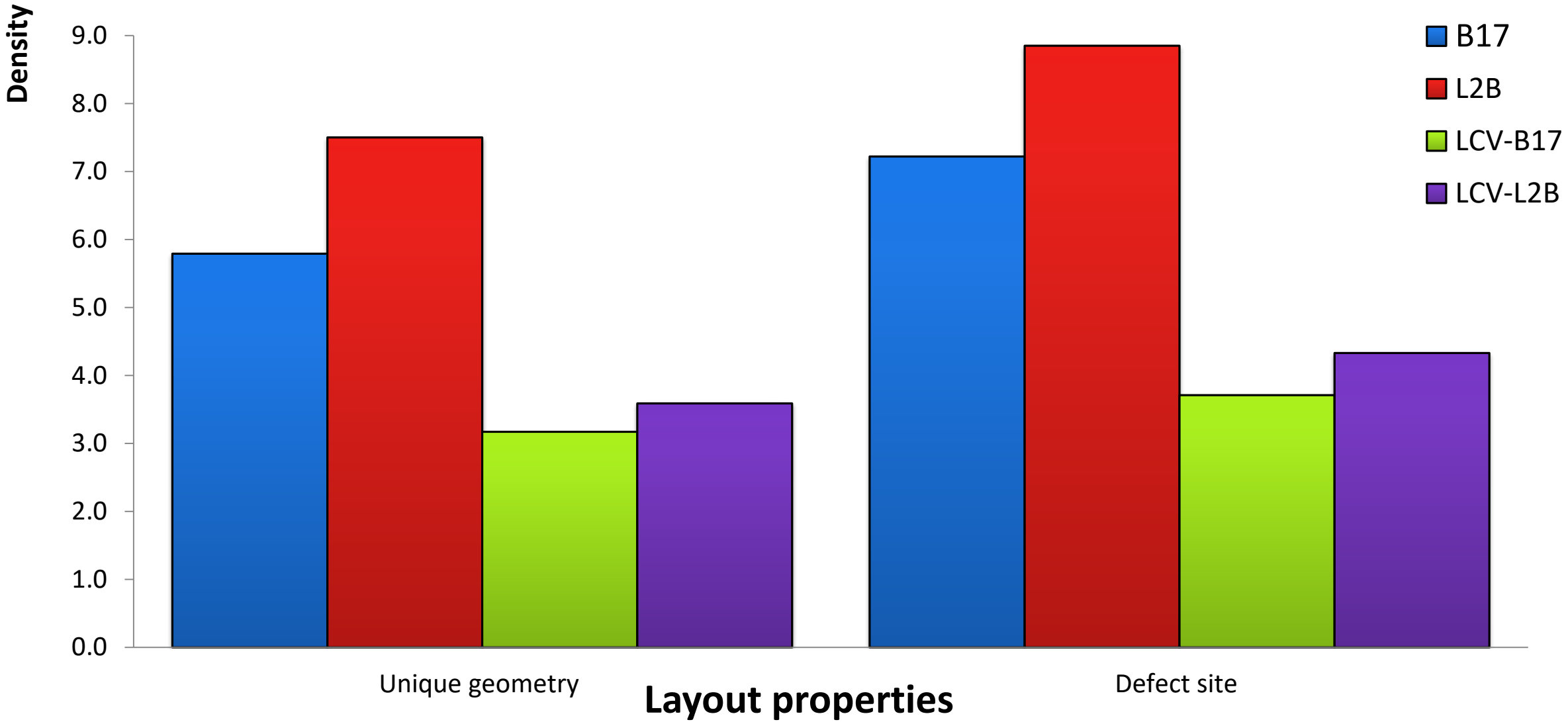
    SSL, bridging, input pattern and path-delay fault coverage
  - Routing complexity:

    Unique geometry and defect site density
- Synthesized ITC and product designs for reflection and comparison
  - b17 and L2B (sub-circuit from OpenSparc T2)
- Designs constructed using 65nm and 45nm cell libraries

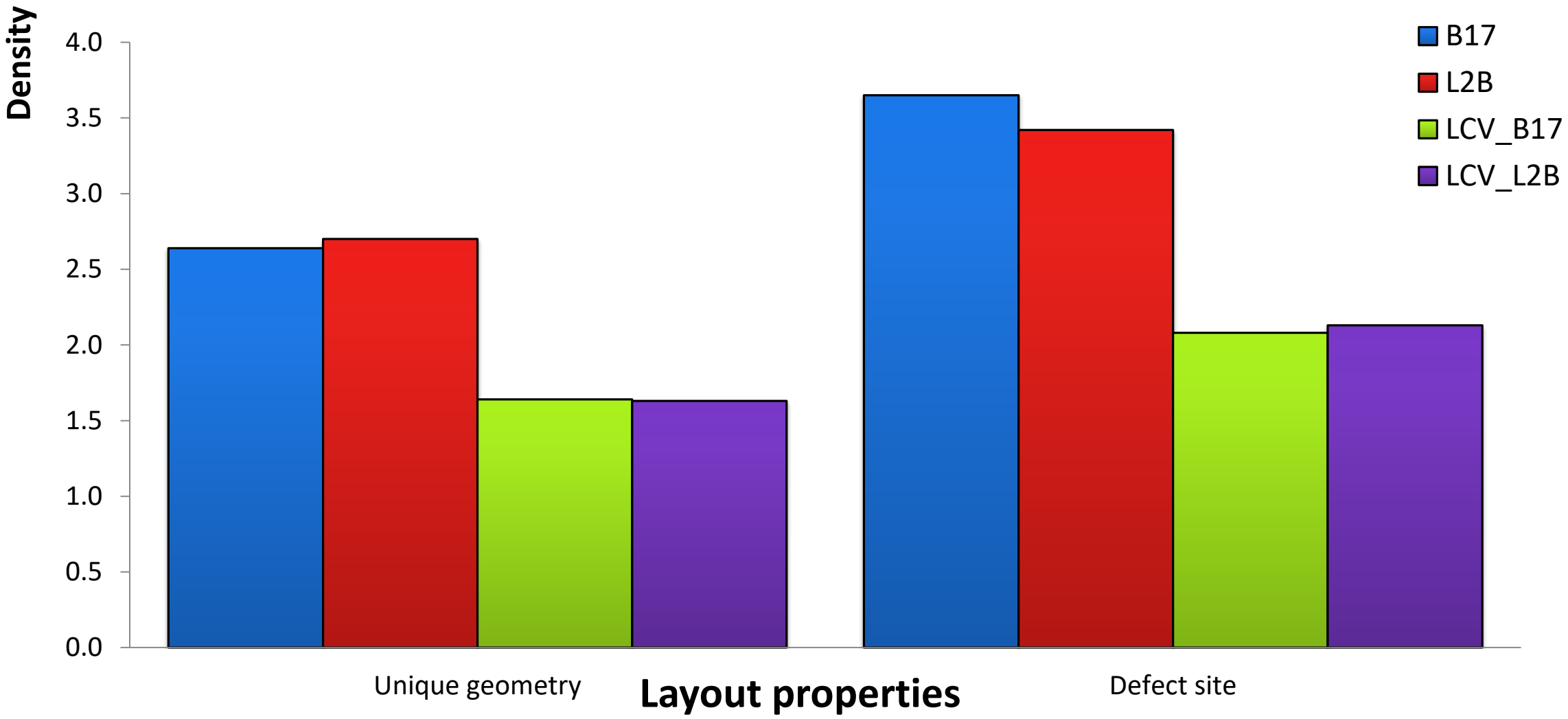
Testability Comparison



Routing Complexity Comparison (45nm)



Routing Complexity Comparison (65nm)



Summary

CM-LCV

- Straightforward, automated design
- Scalable
- High testability and diagnosable
- Demographic reflection
- Silicon dense

Over 60 designs have been fabricated

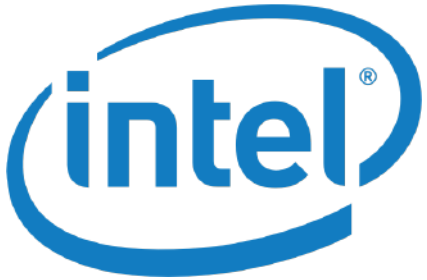
- 14nm AND 7nm
- Have analyzed fail data for two designs

Current Work

1) Improve CM-LCV design efficiency using ML
2) High-dimension FUB array
3) Bijective function exploration
4) Efficient solver for CM-LCV

Acknowledgments

We like to thank our sponsors and collaborators



National Science Foundation
WHERE DISCOVERIES BEGIN

CM-LCV Publications

1. Z. Liu, P. Fynan, and R. D. Blanton, “Improving Test Chip Design Efficiency via Machine Learning,” *IEEE International Test Conference*, Nov. 2019.
2. Z. Liu and R. D. Blanton, “Back-End Layout Reflection for Test Chip Design,” *IEEE International Conference on Computer Design*, Oct. 2018.
3. Z. Liu, P. Fynan, and R. D. Blanton, “Front-End Layout Reflection for Test Chip Design,” *IEEE International Test Conference*, Oct.-Nov. 2017.
4. B. Niewenhuis, S. Mittal and R. D. Blanton, “Multiple-Defect Diagnosis for Logic Characterization Vehicles,” *IEEE European Test Symposium*, May 2017.
5. S. Mittal, Z. Liu, B. Niewenhuis and R. D. Blanton, “Test Chip Design for Optimal Cell-Aware Diagnosability” *IEEE International Test Conference*, Nov. 2016.
6. P. Fynan, Z. Liu, B. Niewenhuis, S. Mittal, M. Strojwas and R. D. Blanton, “Logic Characterization Vehicle Design Reflection via Layout Rewiring,” *International Test Conference*, Nov. 2016.
7. Z. Liu, S. Mittal, B. Niewenhuis and R. D. Blanton, “Achieving 100% Cell-Aware Coverage by Design” *Design, Test and Automation in Europe*, March 2016.
8. R. D. Blanton, Z. Liu and B. Niewenhuis, “Design Reflection for Optimal Test-Chip Implementation,” *International Test Conference*, Oct. 2015.
9. B. Niewenhuis and R. D. Blanton, “Efficient Built-in Self-Test of Regular Logic Characterization Vehicles,” *IEEE VLSI Test Symposium*, April 2015.
10. R. D. Blanton, C. Taylor and B. Niewenhuis, “Logic Characterization Vehicle Design for Maximal Information Extraction for Yield Learning,” *International Test Conference*, Oct. 2014.