Digging Out Proprietary Security Features from Hardware with a Scan Side Channel Attack

Leonid Azriel

Technion – Israel Institute of Technology

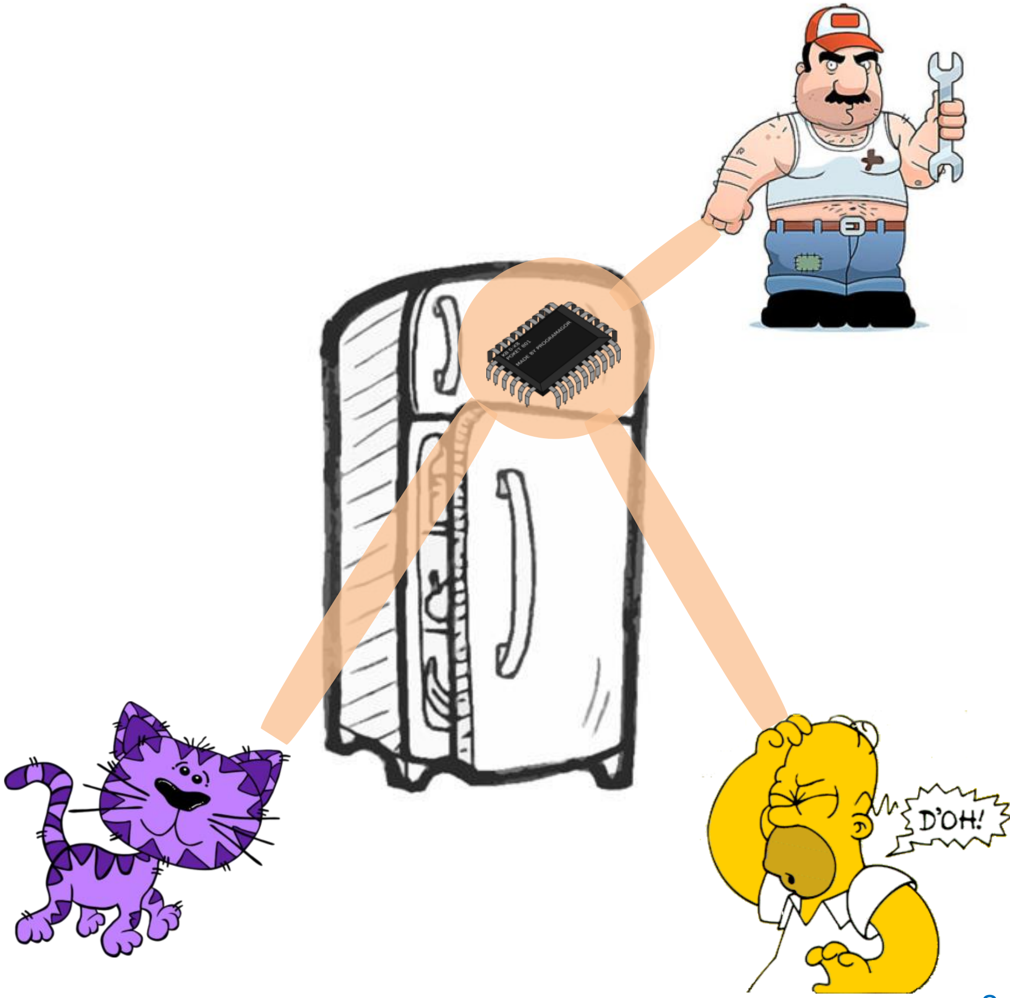
Dec 1, 2015

Research under supervision of Avi Mendelson and Ran Ginosar

IoT Endpoint Security

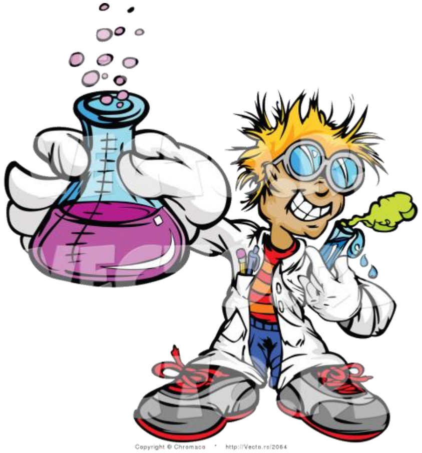
- Internet of **Things**

- Thing = Endpoint
  - Lightweight
  - Privacy concerns
  - Accessible



Reverse Engineering of an ASIC

- Phase 1 – Invasive
  ASIC → Circuit
  - Delayering
  - SEM
  - Nanoscale Imaging
  - Cross-section



- Phase 2 – Algorithmic
  Circuit → Spec
  - FSM Extraction
  - Model Checking
  - SAT

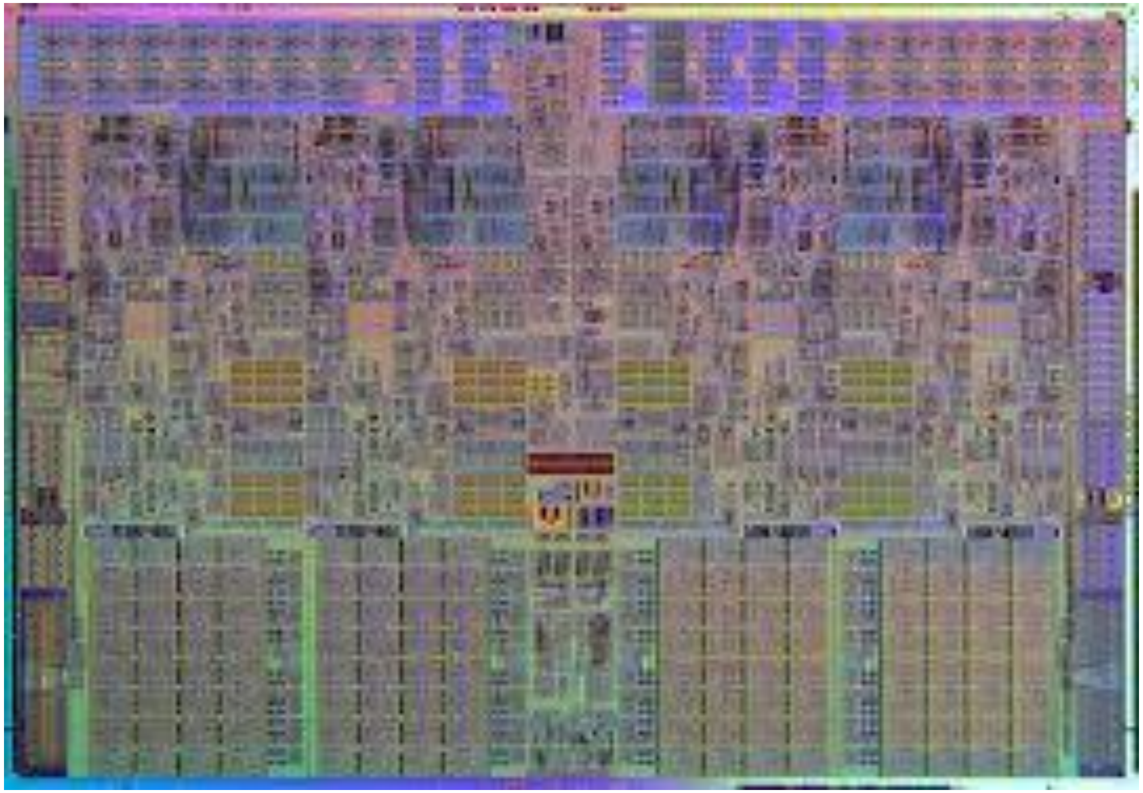
Reverse Engineering of an ASIC

- Phase 1 – Invasive
  ASIC → Circuit
  - Delayering
  - SEM
  - Nanoscale Imaging
  - Cross-section

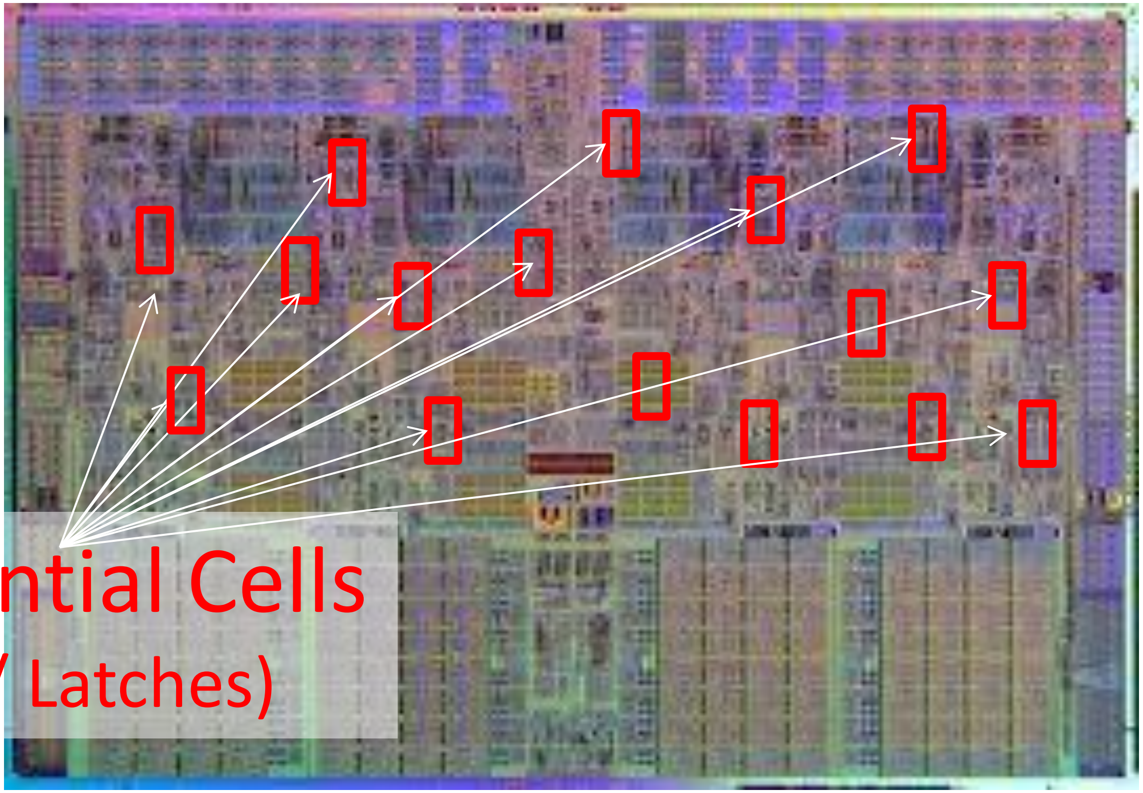
- Phase 2 – Algorithmic
  Circuit → Spec
  - FSM Extraction
  - Model Checking
  - SAT Solvers

Scan Side Channel makes phase 1 non-invasive

The Scan Technique



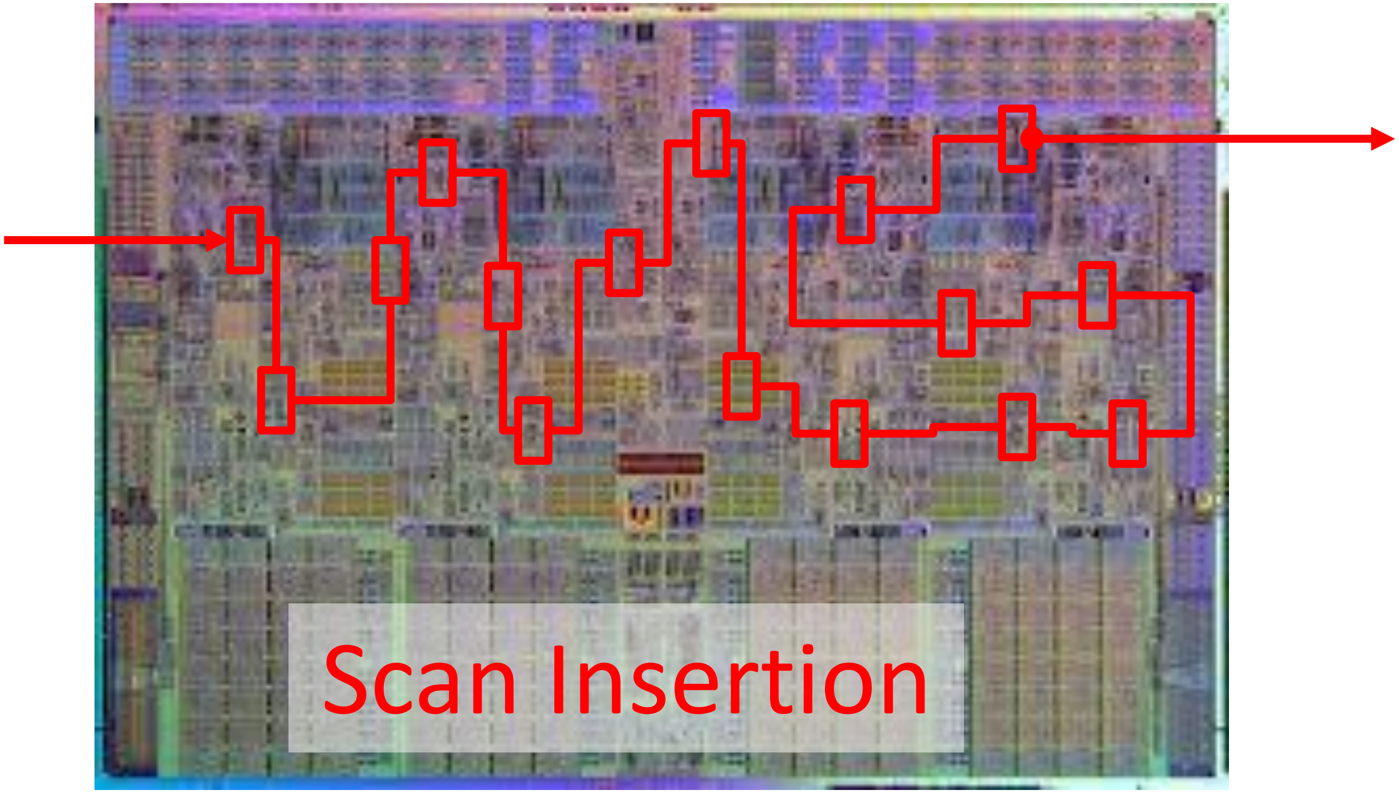
The Scan Technique



Sequential Cells

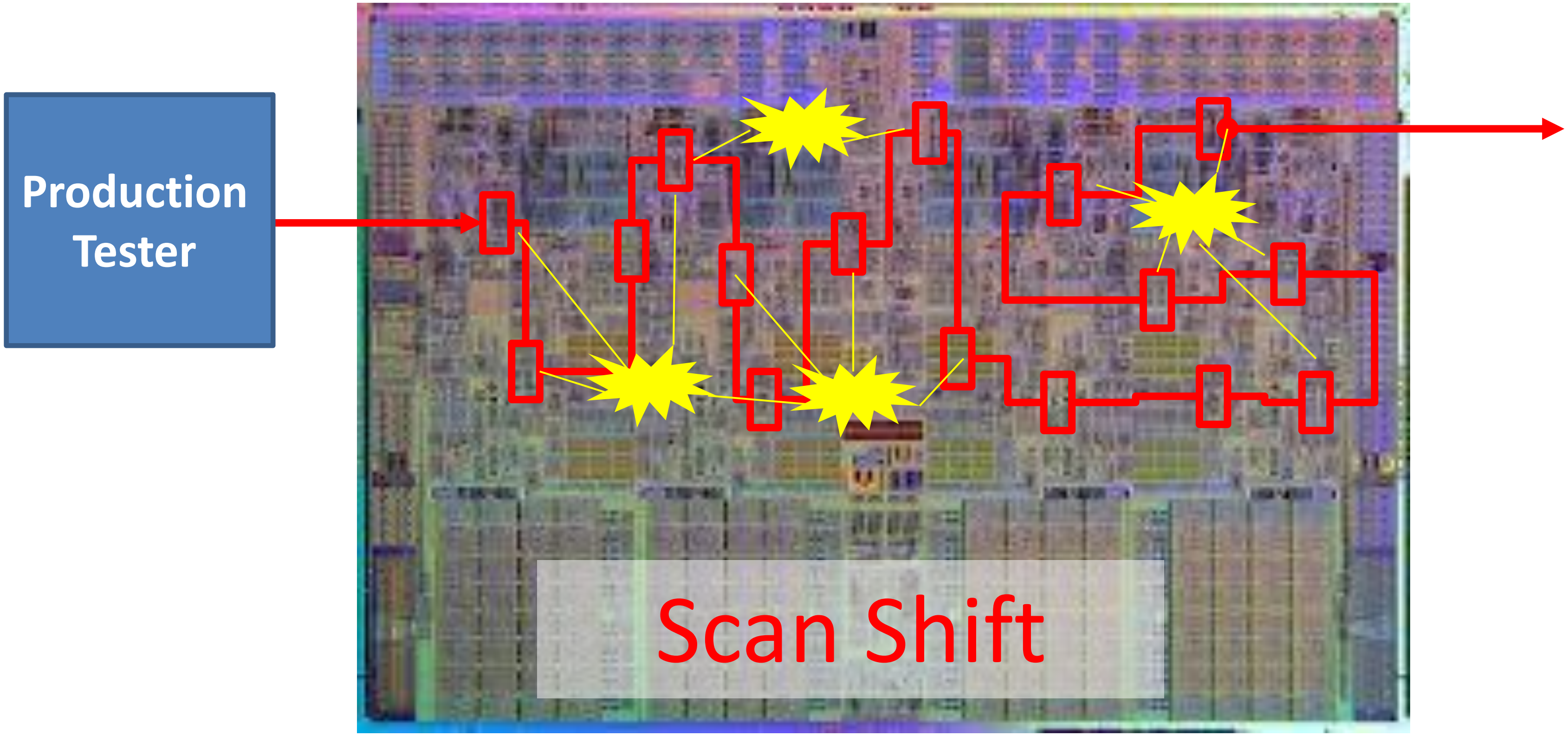
(FFs / Latches)

The Scan Technique

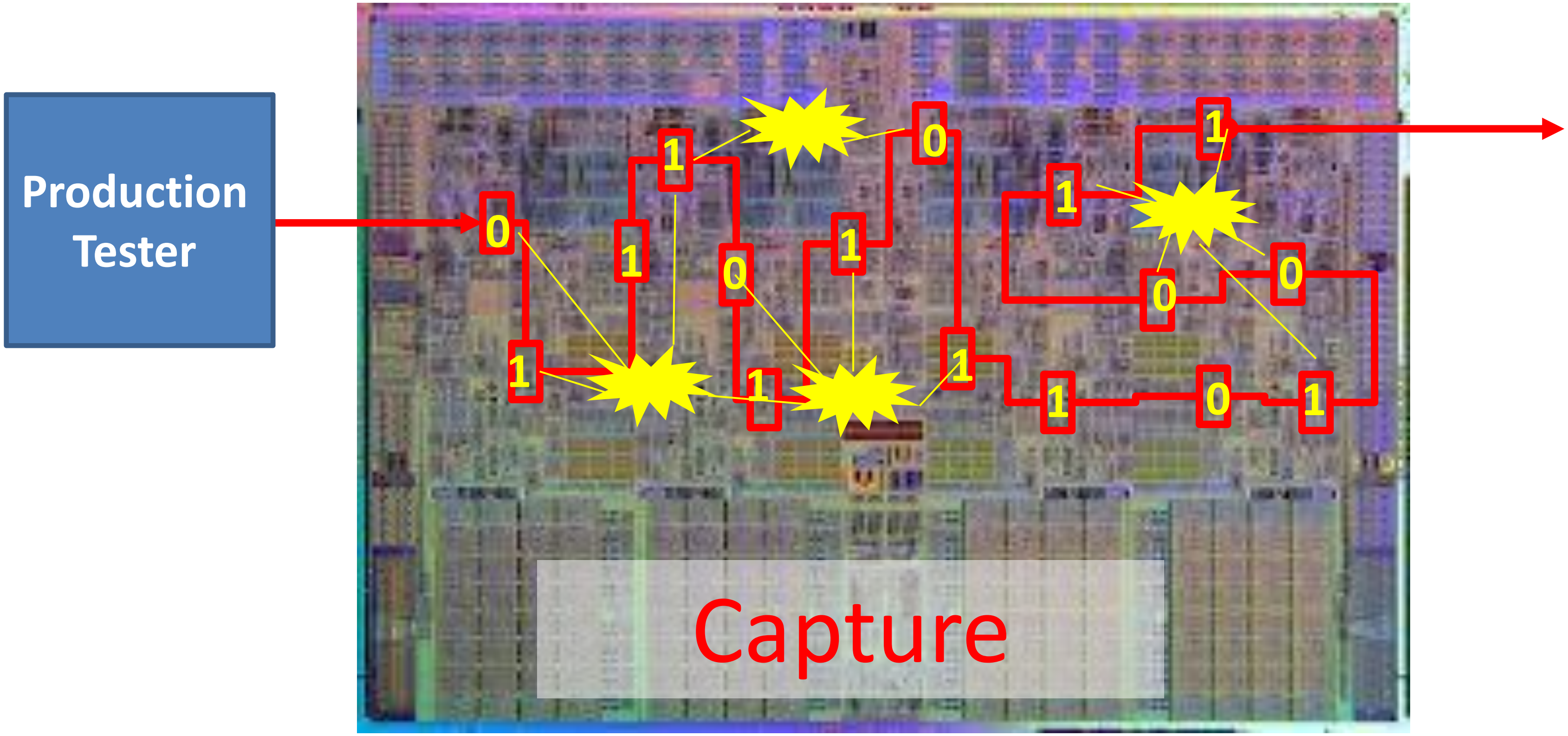


Scan Insertion

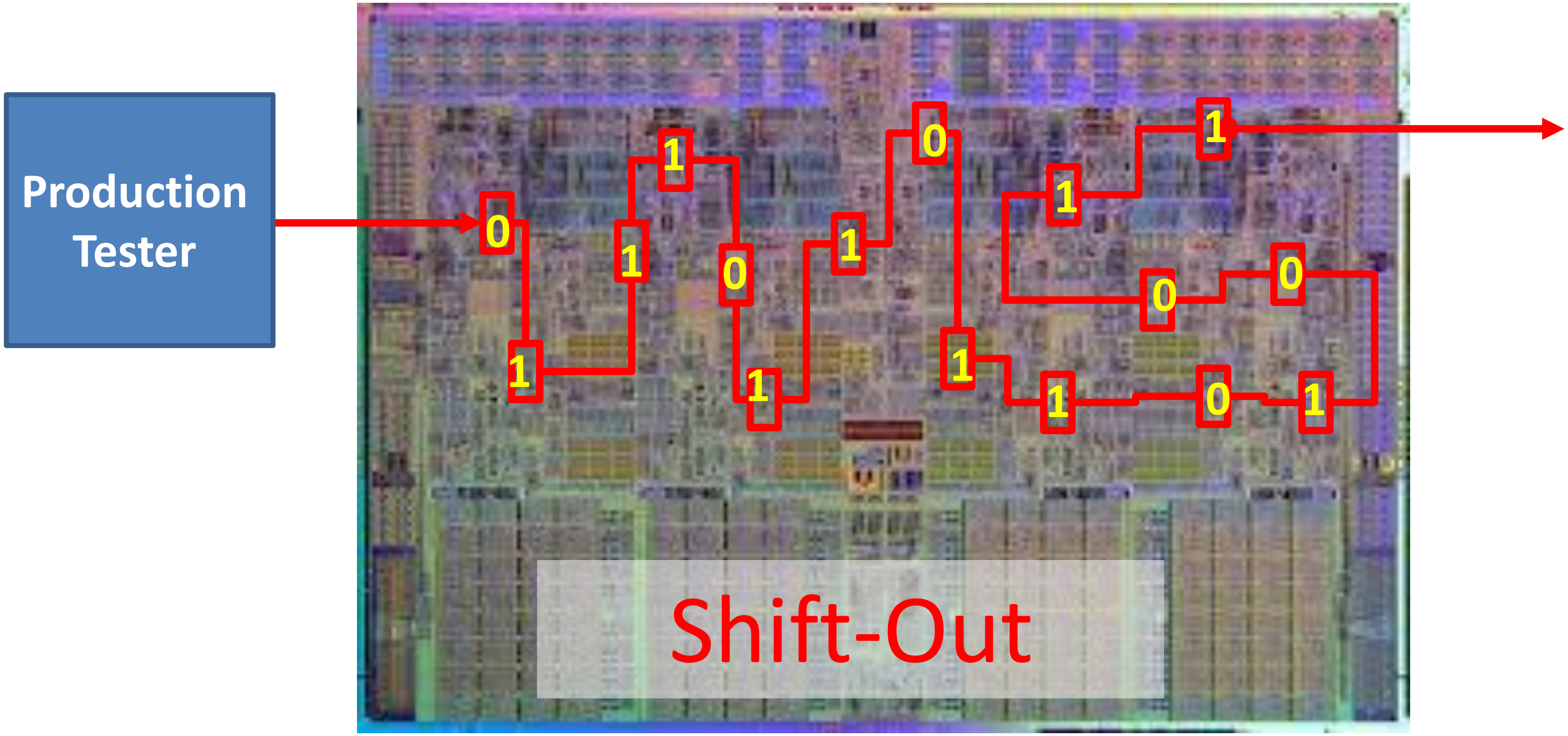
The Scan Technique



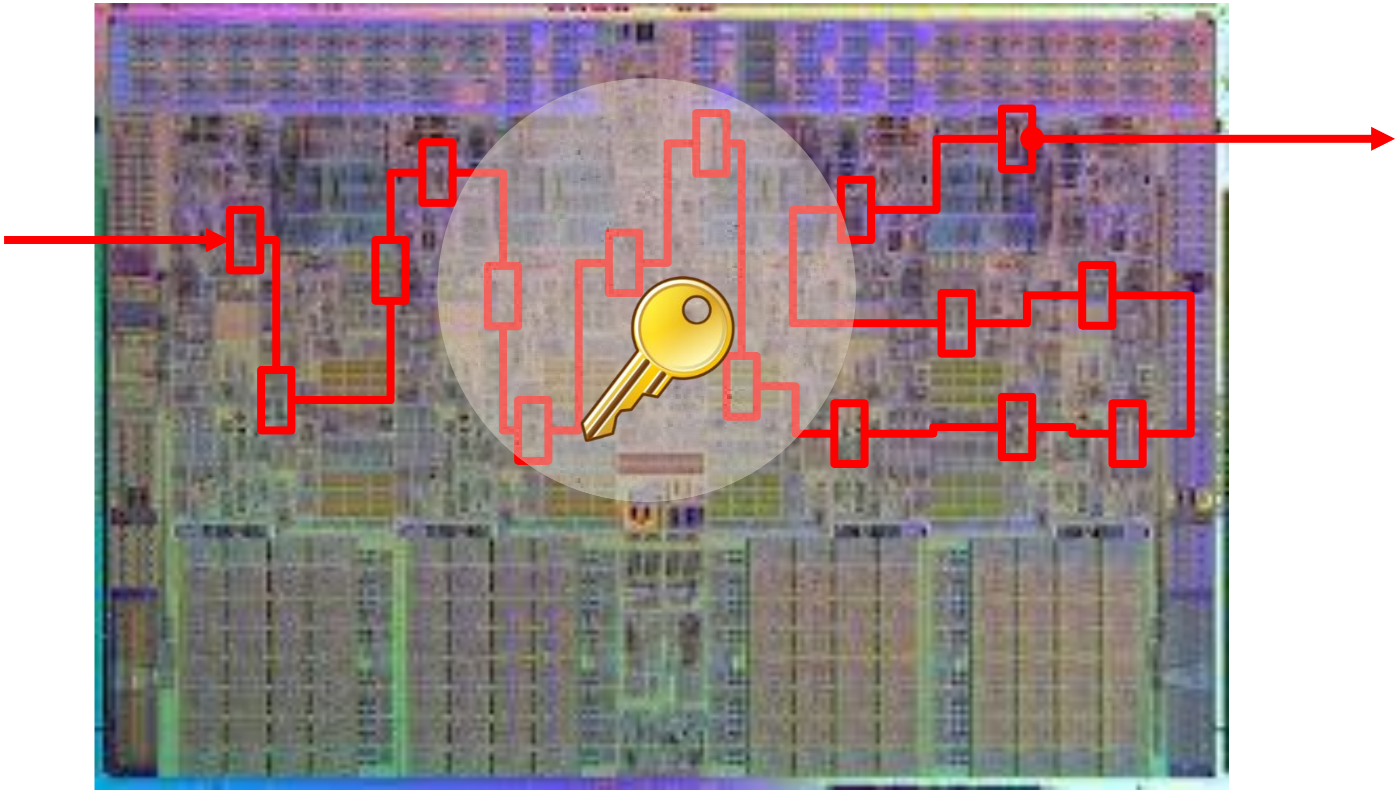
The Scan Technique



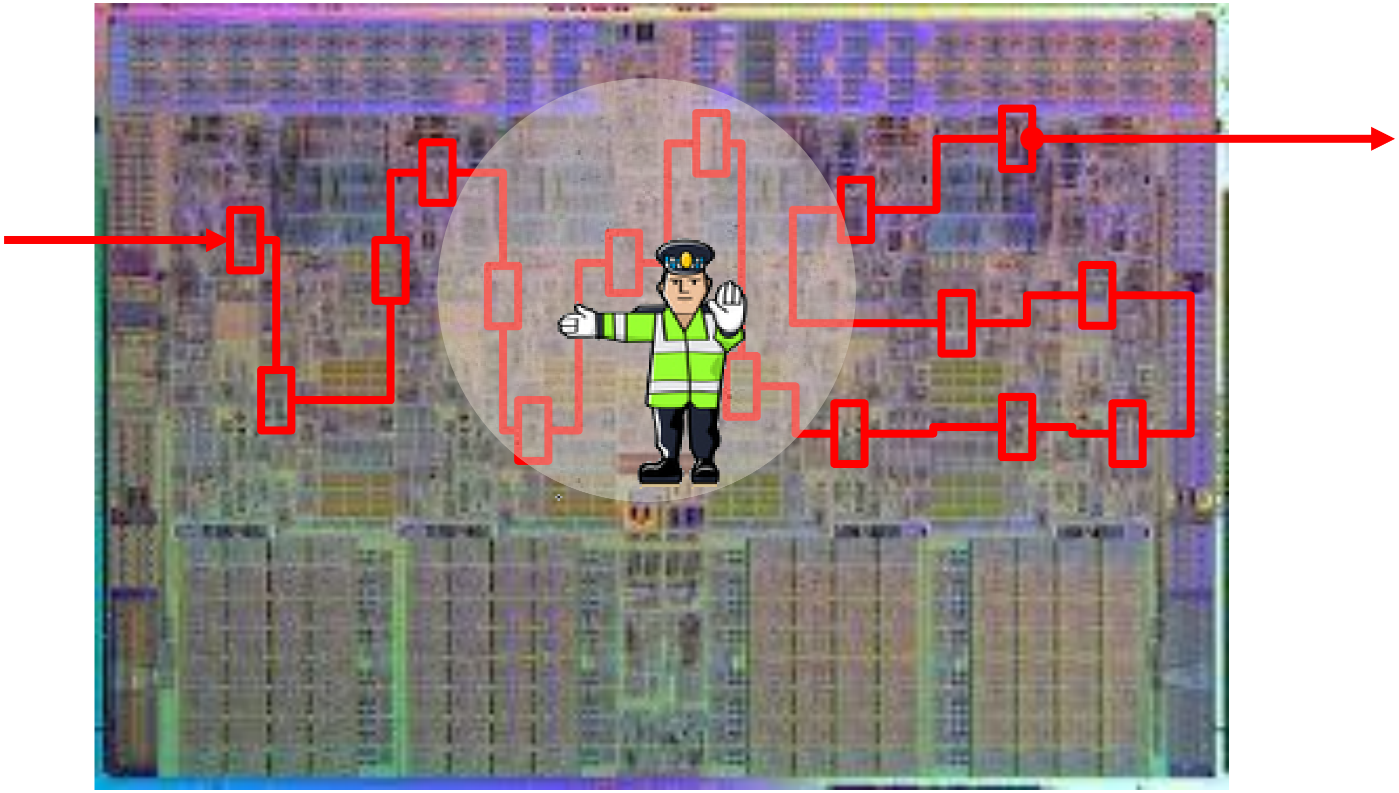
The Scan Technique



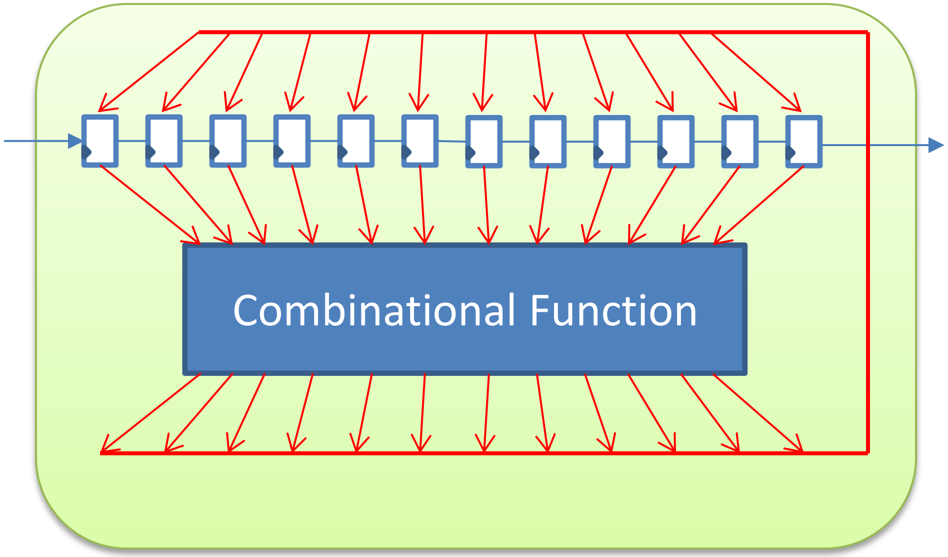
Exploiting Scan - Retrieving Secrets



Exploiting Scan – Altering the Flow

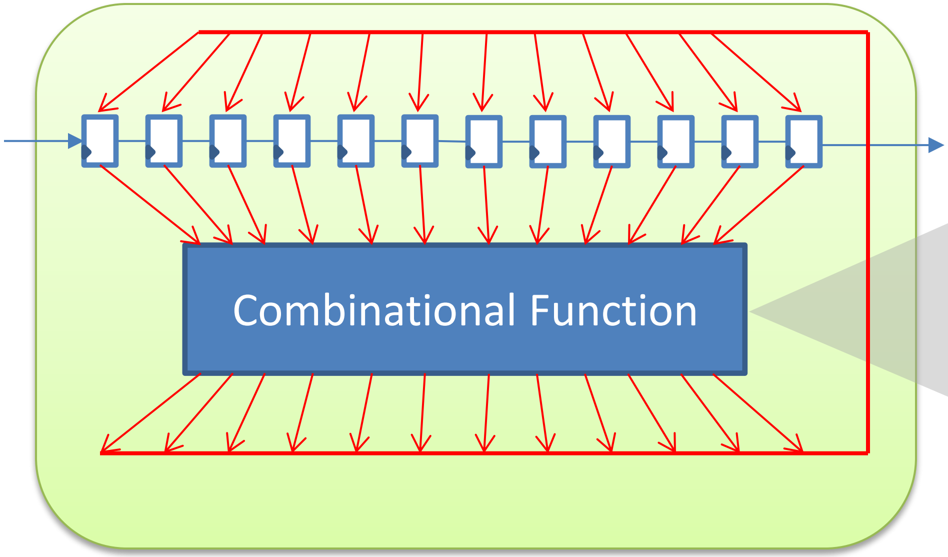


Unfolding Sequential Circuits with Scan



- Scan turns the ASIC to a stateless circuit
- Mapped to the Boolean Function Learning problem: $\{0,1\}^n \rightarrow \{0,1\}^n$

Unfolding Sequential Circuits with Scan

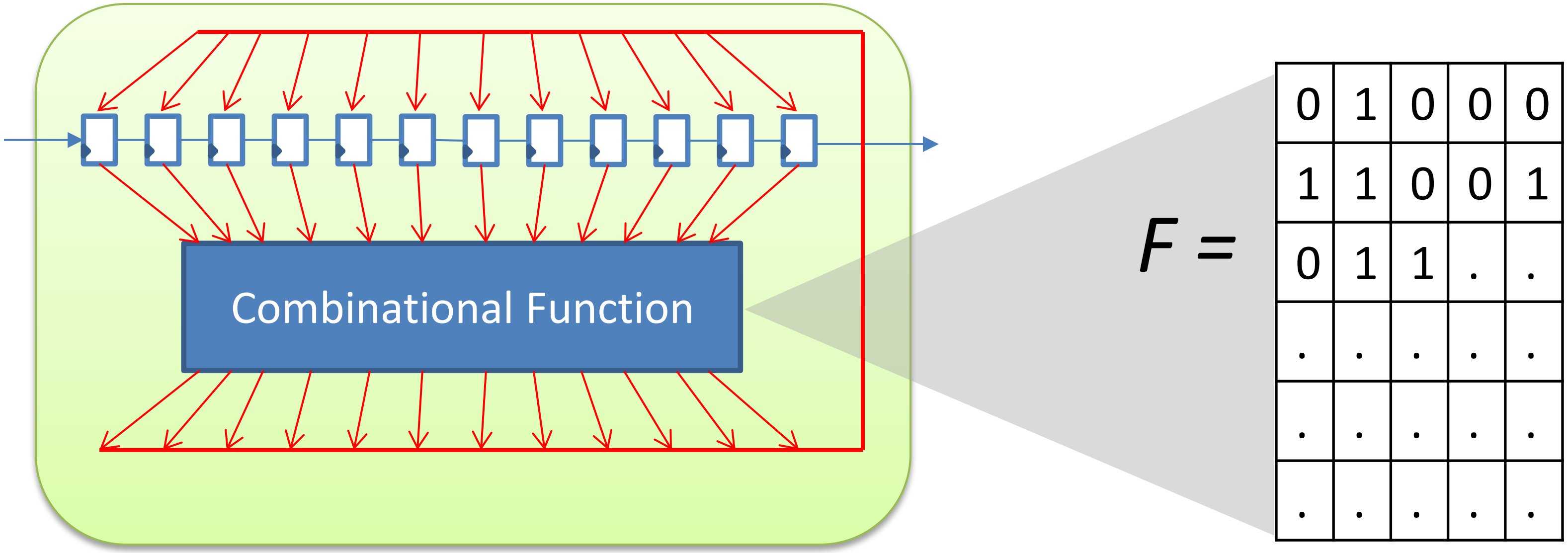


$F =$

0	1	0	0	0
1	1	0	0	1
0	1	1	.	.
.	.	.	.	.
.	.	.	.	.
.	.	.	.	.

- Scan turns the ASIC to a stateless circuit
- Mapped to the Boolean Function Learning problem: $\{0,1\}^n$ → $\{0,1\}^n$
- Exhaustive Search: Extract the Truth Table by running queries for all inputs

Unfolding Sequential Circuits with Scan



$F =$

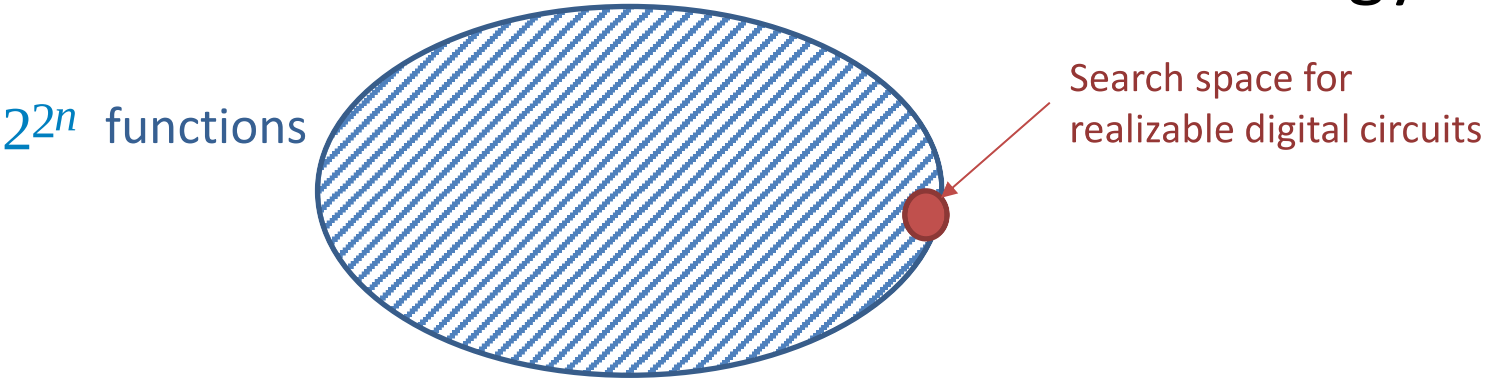
0	1	0	0	0
1	1	0	0	1
0	1	1	.	.
.	.	.	.	.
.	.	.	.	.
.	.	.	.	.

- Scan turns the ASIC to a stateless circuit
- Mapped to the Boolean Function Learning problem: $\{0,1\}^n \rightarrow \{0,1\}^n$
- Exhaustive Search: Extract the Truth Table by running queries for all inputs
- Exponential Size: 2^n

Shannon Effect

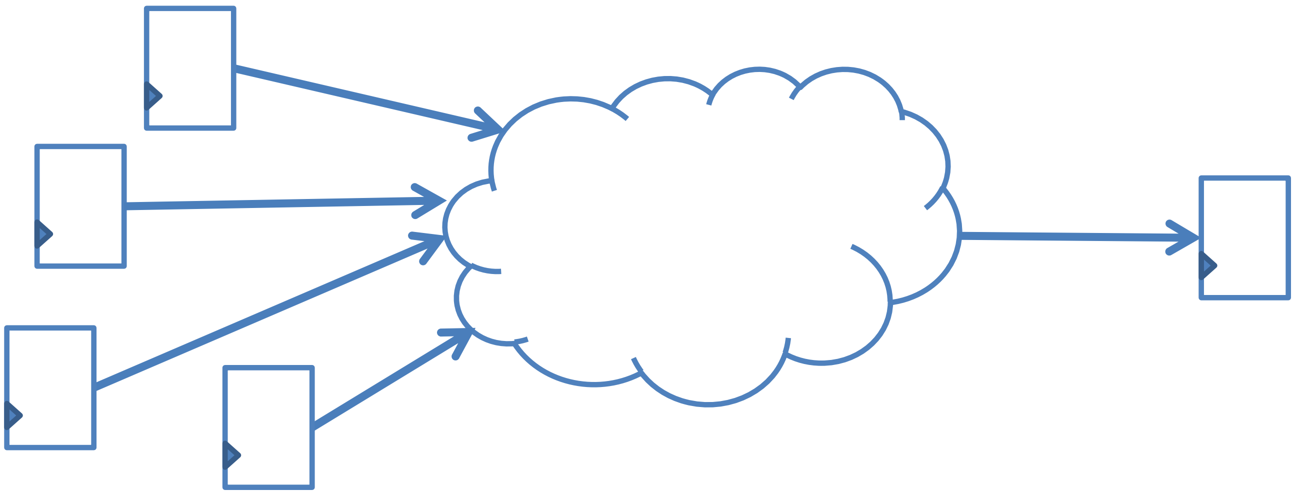
- Shannon Effect: “almost all” Boolean functions have a complexity close to the maximal possible (~$O(2^n)$) for the uniform probability distribution

- Corollary: For large n, “almost all” Boolean functions are not realizable in VLSI technology



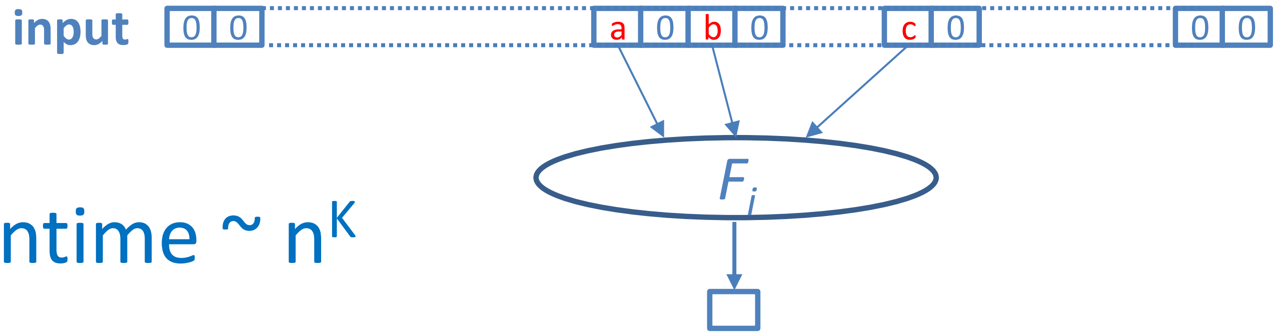
Limited Transitive Fan-in

- In practice, logic cones have limited number of inputs: Transitive Fan In = K



Algorithm for Limited Transitive Fan-in

- Suppose *F*(0) = 0 (simple extension to any *F*)
- Example for K = 3:
  - Testing all values of input vector with Hamming Weight 3 or less covers all combinations of *{a,b,c}*



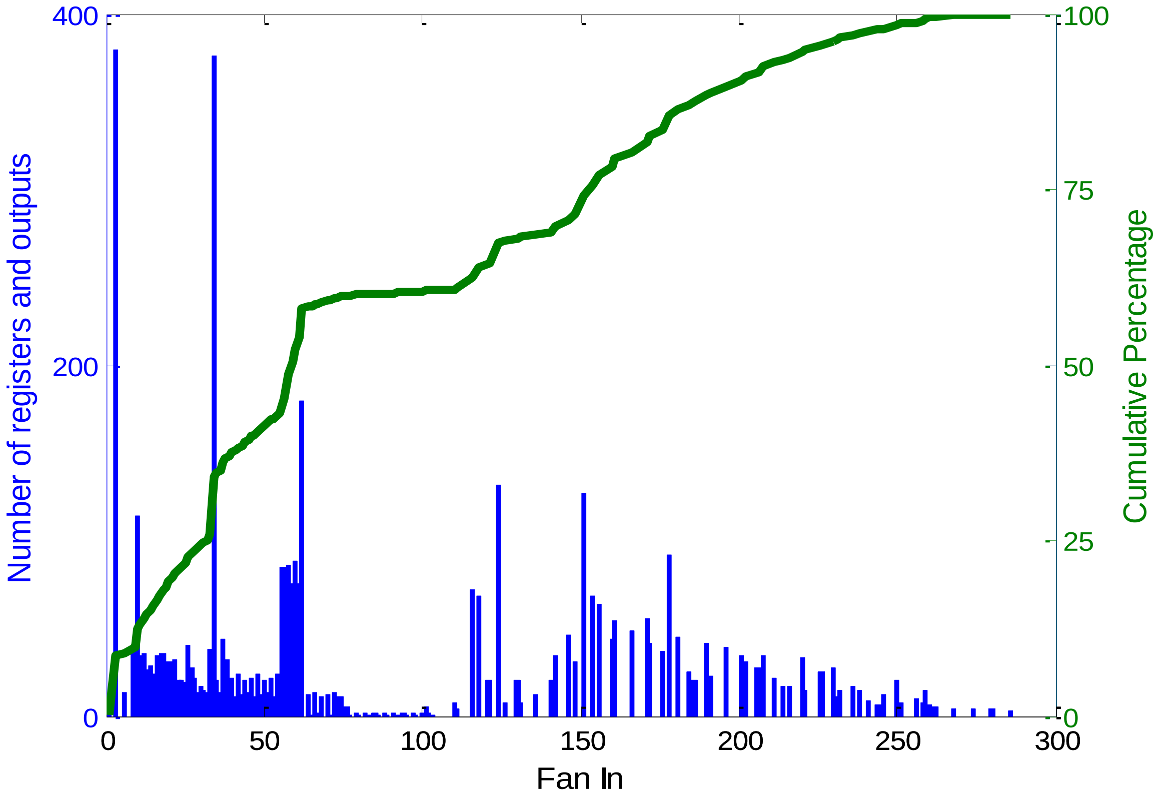
- Runtime ~ n^K

Junta Learning

1	0	1	1	1	0	0	1	0	1	1	0	1	0	1	-	-	-	-	1	1
1	1	0	1	1	0	0	1	0	1	0	0	1	0	1	-	-	-	-	1	0
0	1	1	0	1	1	0	0	0	0	0	1	1	0	1	-	-	-	-	0	1
0	1	0	1	0	1	1	0	1	1	0	0	0	0	0	-	-	-	-	0	0
0	0	1	1	0	0	1	1	0	1	1	1	0	1	1	-	-	-	-	0	1
1	1	1	1	1	0	1	1	1	1	0	0	0	1	0	-	-	-	-	0	1
0	0	1	1	0	1	0	0	1	0	1	1	1	1	1	-	-	-	-	0	1
1	0	1	0	1	1	0	1	1	0	0	0	0	0	0	-	-	-	-	1	0
1	1	0	0	0	0	1	1	0	0	0	1	0	0	1	-	-	-	-	1	0
0	1	0	1	1	0	1	0	1	1	0	1	0	1	1	-	-	-	-	1	0
0	1	0	1	0	1	0	0	0	0	1	1	1	0	0	-	-	-	-	0	1
0	1	1	0	1	1	1	0	1	1	1	0	0	1	0	-	-	-	-	1	0
0	1	0	0	1	1	0	1	1	0	0	0	1	1	0	-	-	-	-	1	1
1	1	0	1	1	1	0	1	0	0	0	1	1	0	1	-	-	-	-	1	1
1	1	0	0	1	0	1	0	1	0	1	1	0	0	0	-	-	-	-	0	0
1	0	1	1	0	0	1	1	0	0	1	0	0	0	1	-	-	-	-	0	1
0	1	0	1	0	0	0	0	0	0	0	0	1	0	0	-	-	-	-	0	0
0	1	1	1	0	1	1	1	0	1	1	0	0	0	0	-	-	-	-	1	0
0	1	0	1	1	1	1	1	1	1	1	0	1	1	0	-	-	-	-	0	1
0	0	1	0	1	1	1	0	1	1	0	0	1	0	0	-	-	-	-	1	1
1	0	0	1	0	0	1	0	1	1	0	1	0	0	0	-	-	-	-	0	0

Runtime ~ $\underline{2^K}$ ➔ scalable with the chip size

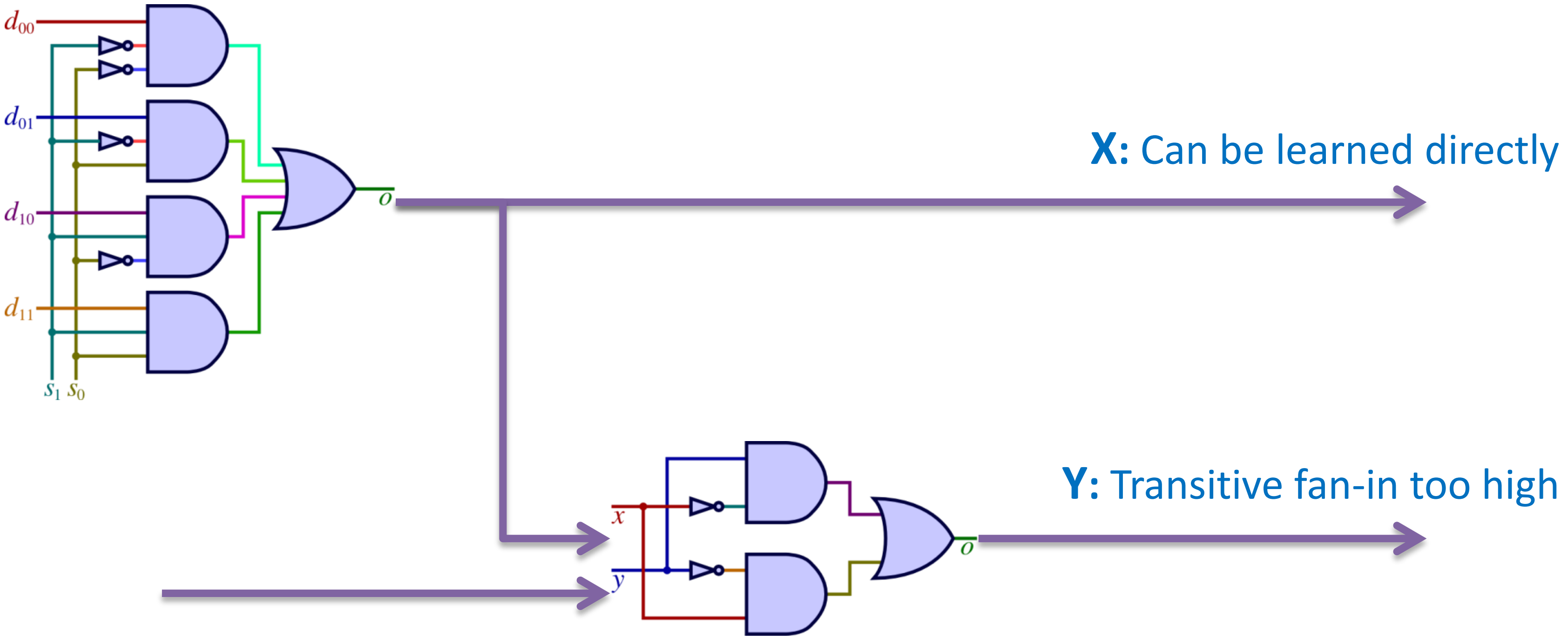
Transitive Fan-in for ITC'99 benchmark



Locality

- Hierarchical structure – loose connectivity between blocks: clustering
- Physical locality: adjacent registers in the chain are likely to belong to the same function
- Often the same sub-circuit is shared by a few logic cones

Sharing sub-circuits



X: Can be learned directly

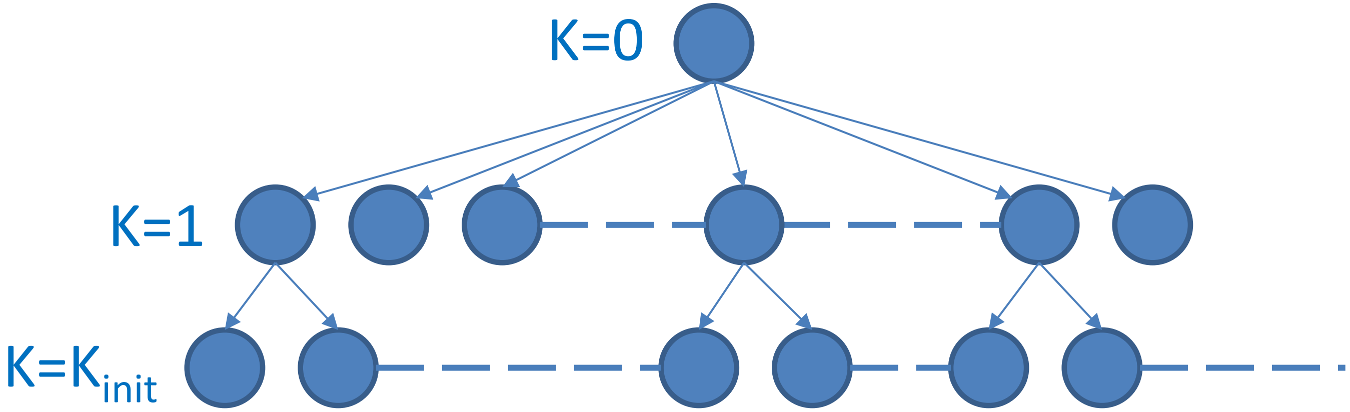
Y: Transitive fan-in too high

X = ab’cde + adf’ + bf’g + ….

Y = ab’cdeg’h + adf’cg + bf’g + ….

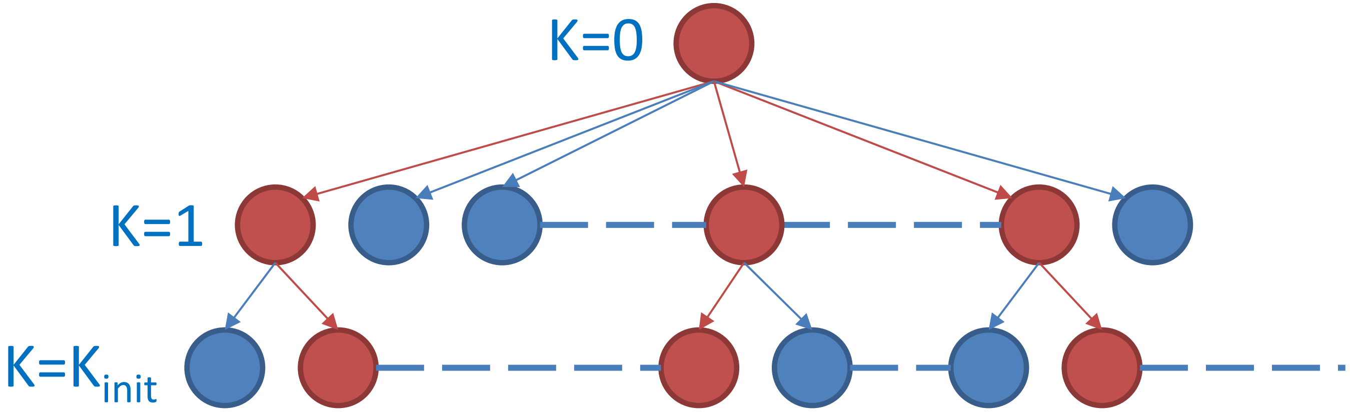
Y is a DNF extension of X

Incremental K-Bounded Search



● = Boolean cube

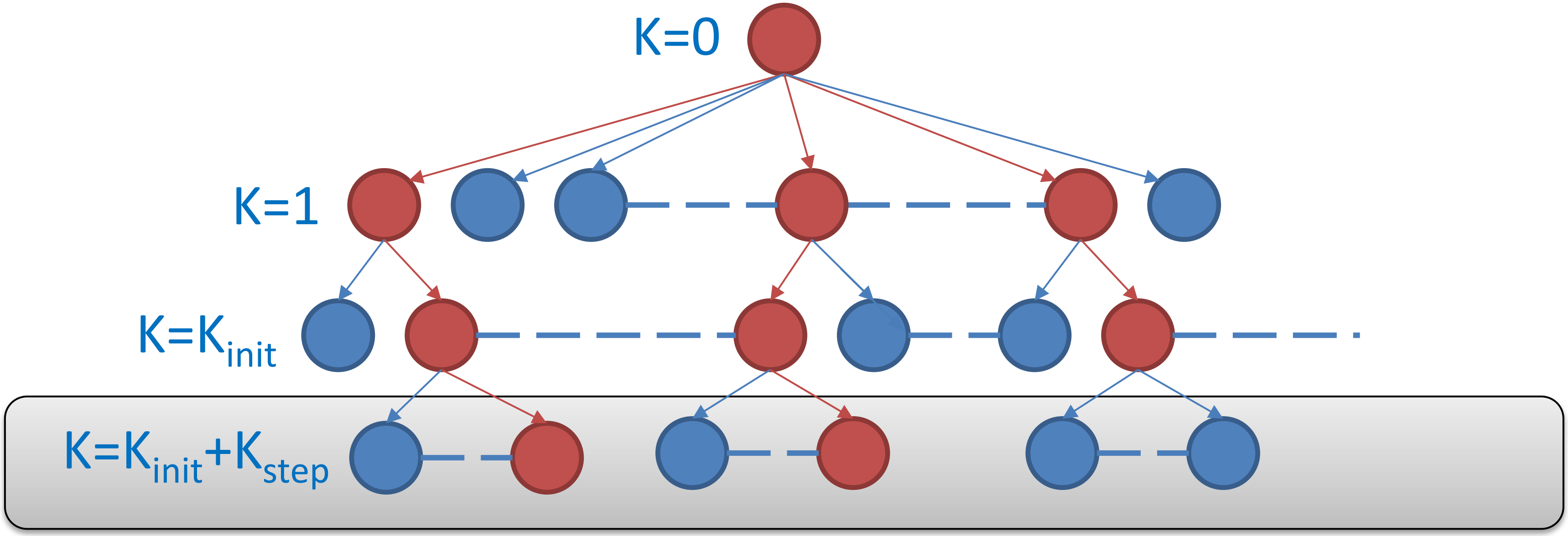
Incremental K-Bounded Search



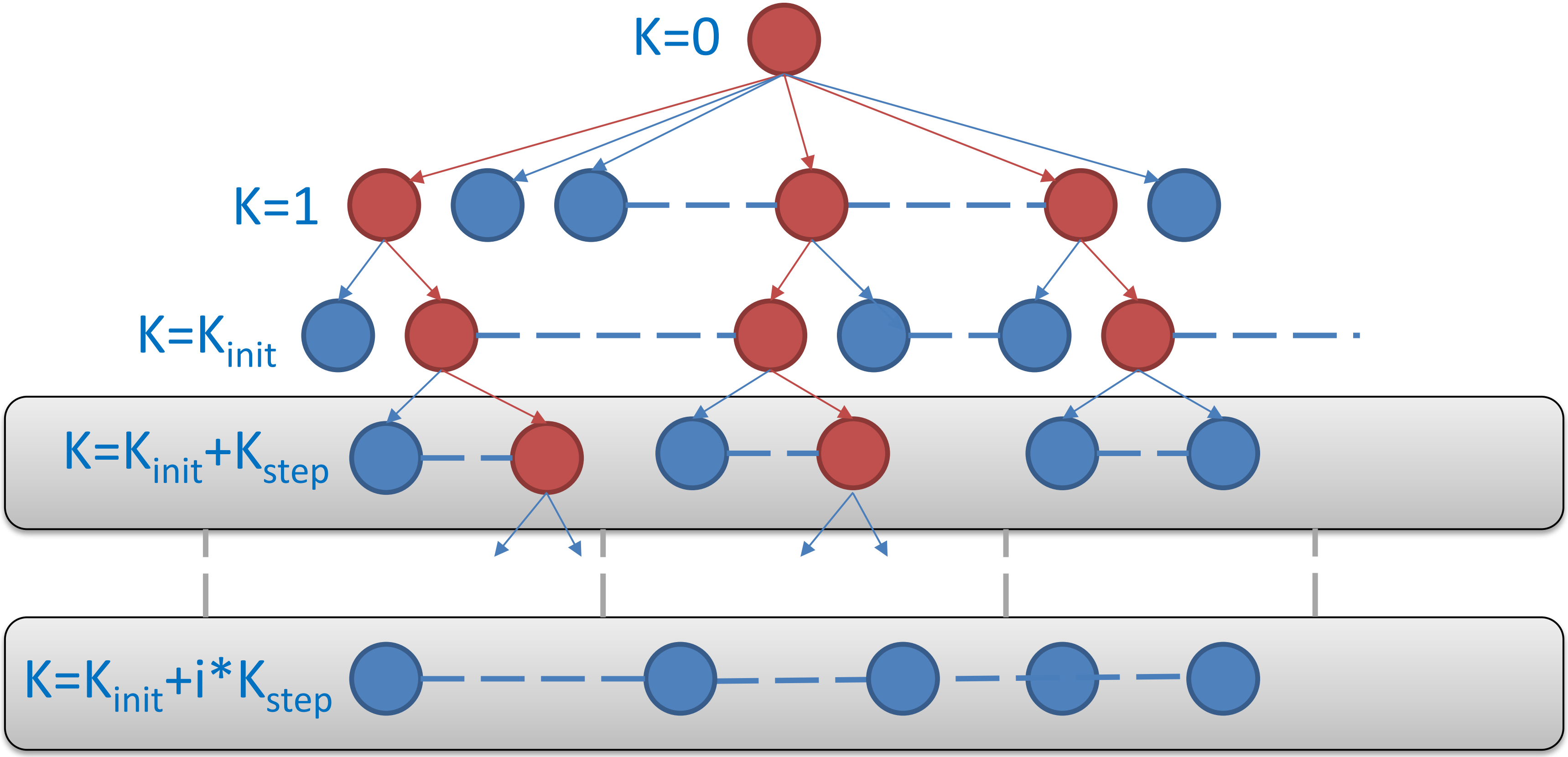
🔵 = Boolean cube

🔴 = Implicant: a cube, for which $F_i=1$ for some *i*

Incremental K-Bounded Search

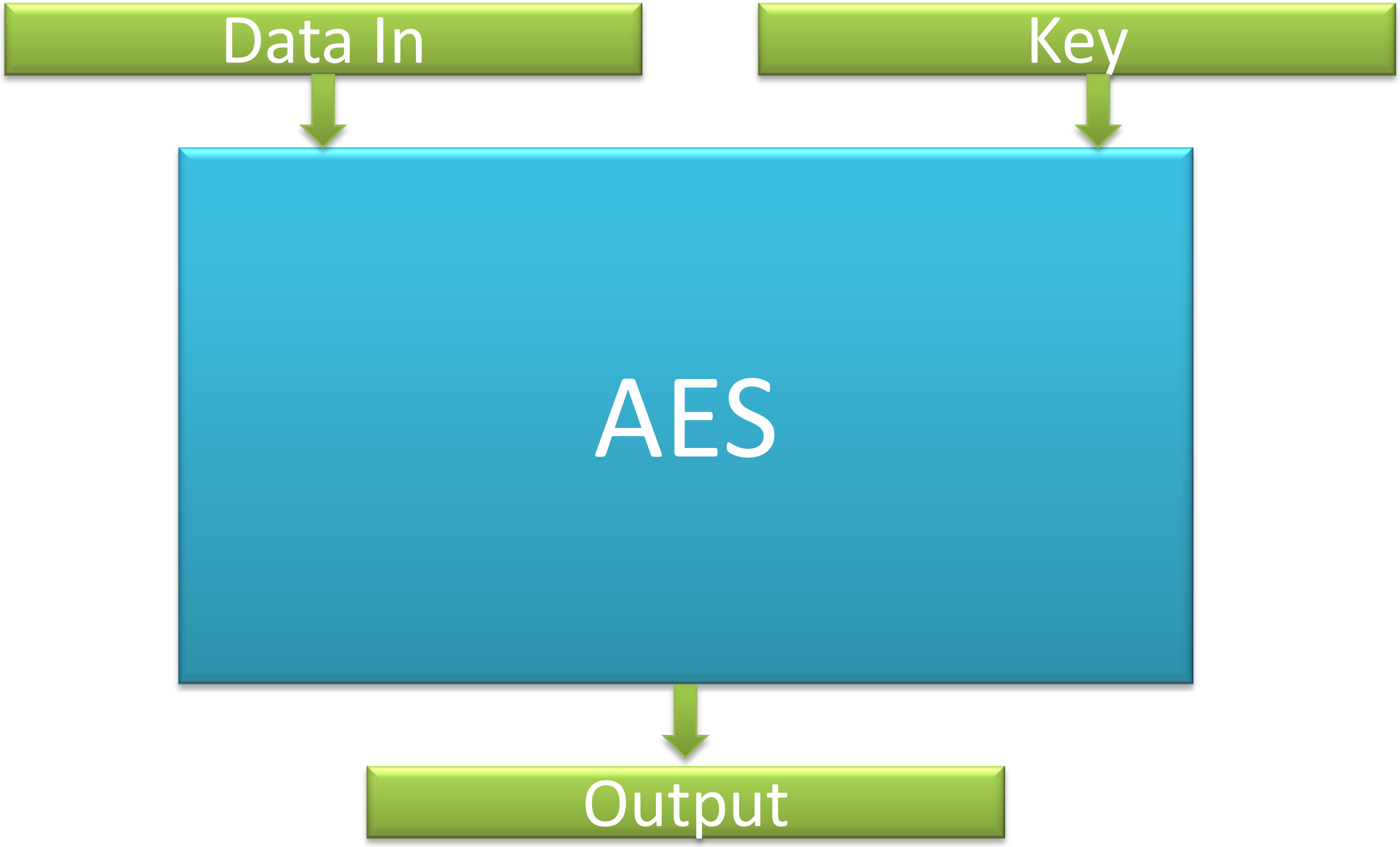


Incremental K-Bounded Search

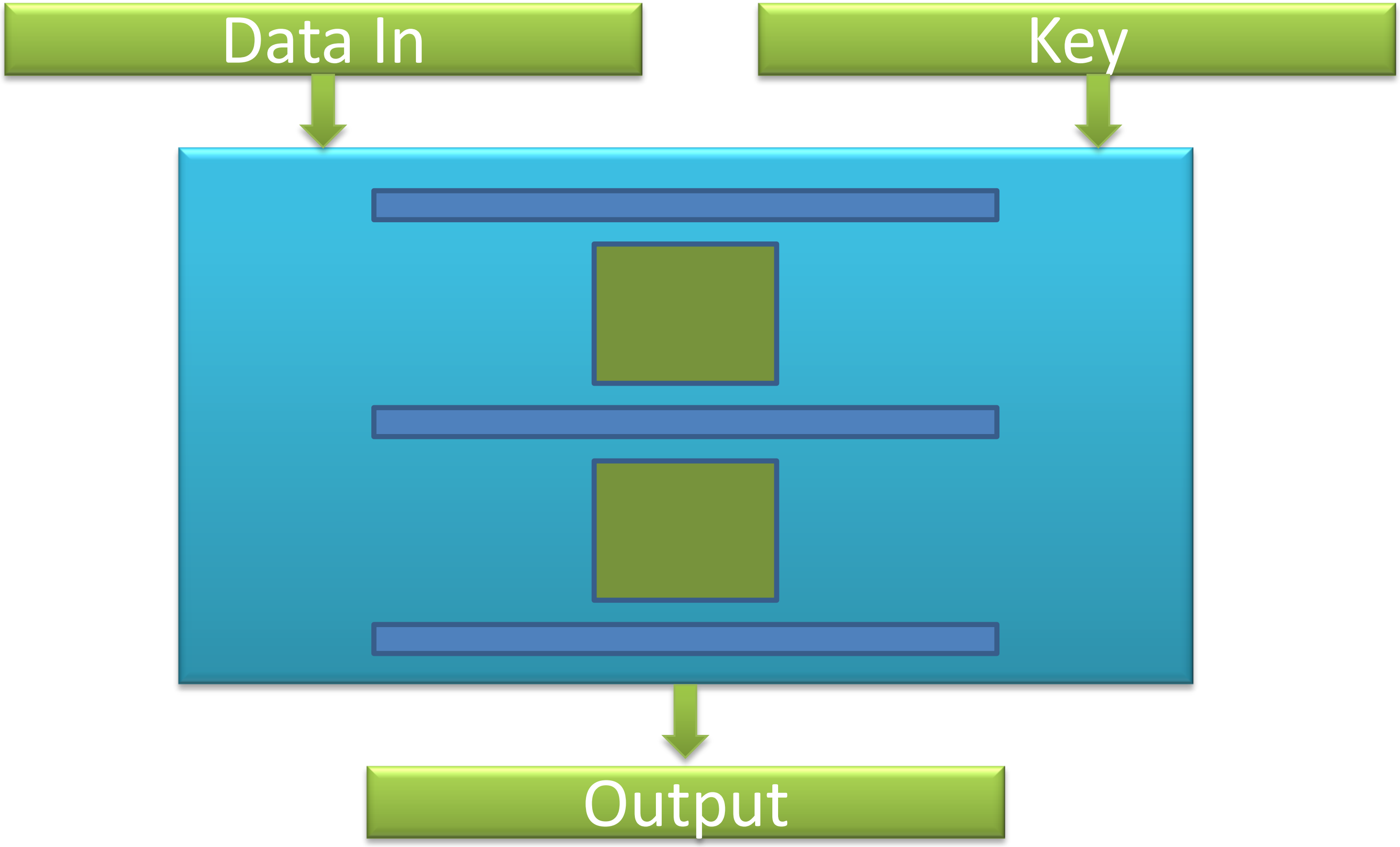


Continue while there is a change

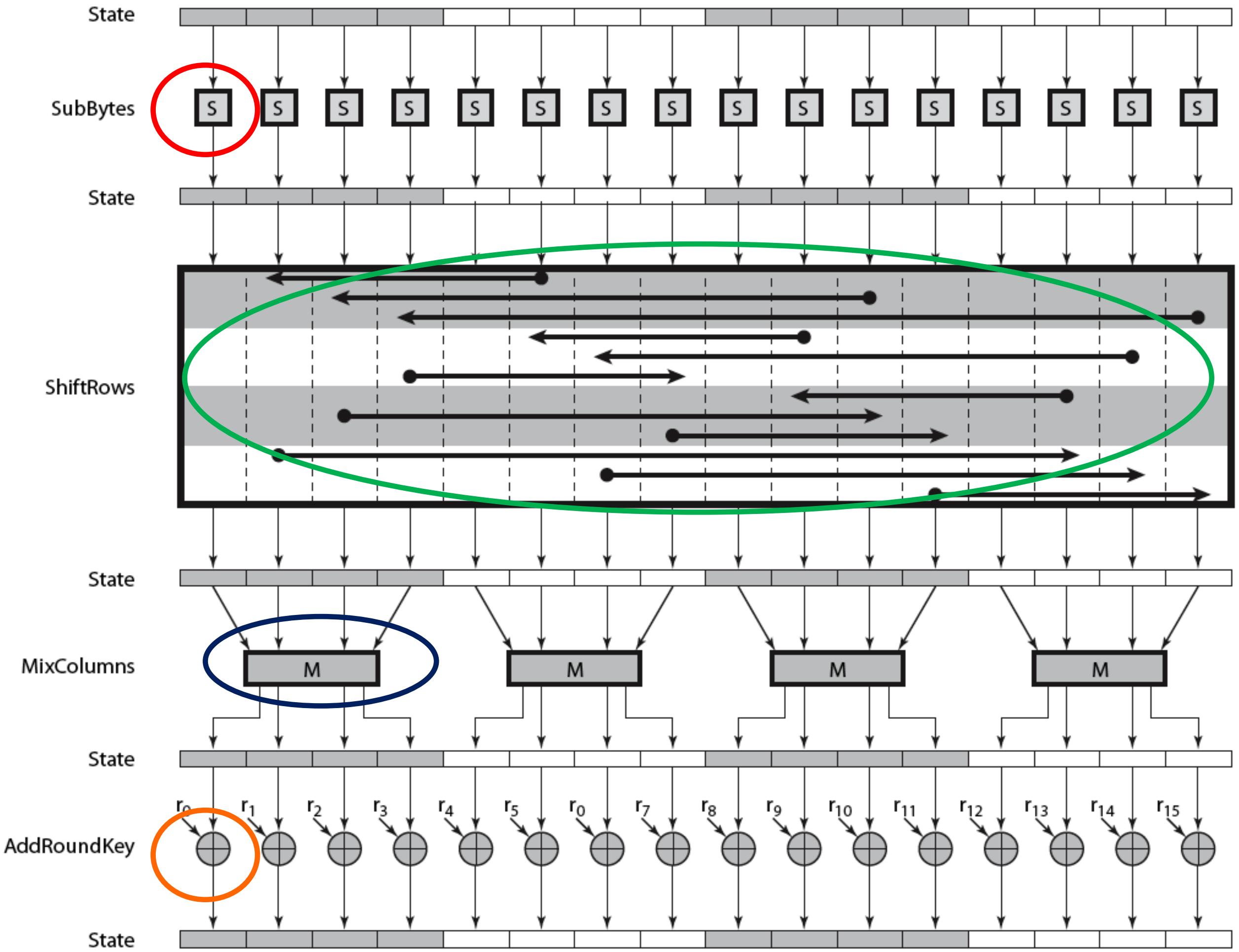
Example: AES



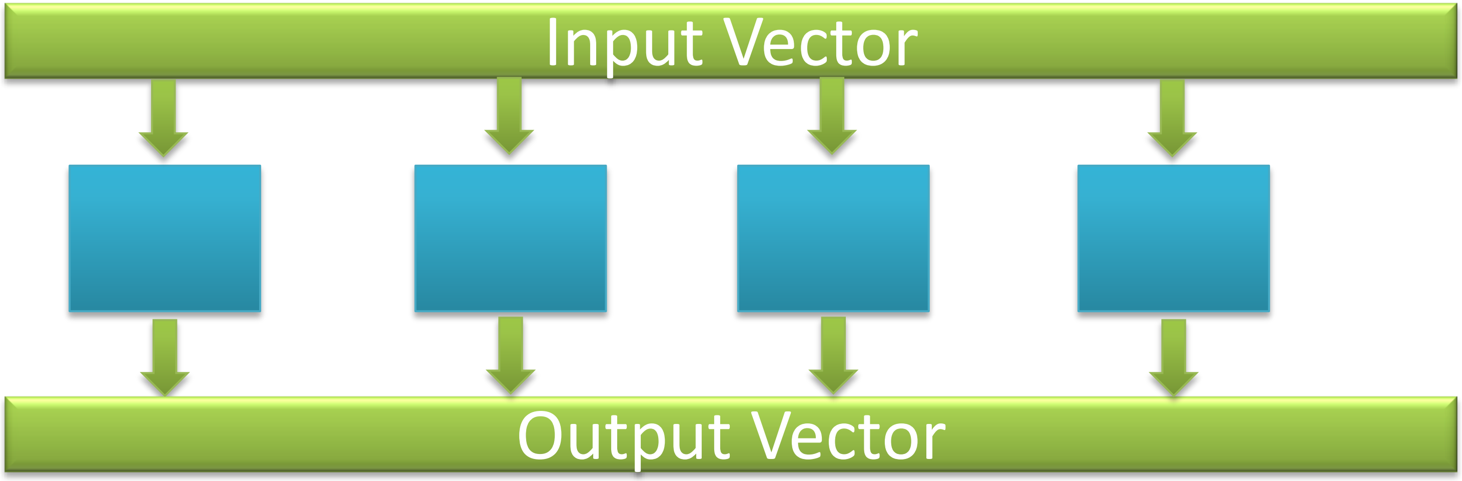
Example: AES



Example: AES


State
SubBytes
S S S S S S S S S S S S S S S S
State
ShiftRows
State
MixColumns
M M M M
State
AddRoundKey
r_0 r_1 r_2 r_3 r_4 r_5 r_0 r_7 r_8 r_9 r_{10} r_{11} r_{12} r_{13} r_{14} r_{15}
State

Example: AES



Learned the Open Cores 'Tiny AES' implementation containing ~8000 registers with only ~1.6M probe operations

- Thanks to the 'avalanche' effect

Countermeasures

- Giving up on scan
- Disabling scan by burning fuses after production
- Logic BIST
- Not allowing dynamic switching
- Protected entry to scan mode

Main Messages

- Reverse Engineering can be non-invasive
- Scan Side Channel is a threat both to security and to IP protection
- Conventional protection methods not always efficient against reverse engineering
  - Need protection targeted to this attack

Thanks!