DISTROY: Detecting IC Trojans with Compressive Measurements

Youngjune Gwon, H. T. Kung, and Dario Vlah
Harvard University

August 9, 2011
Understanding Modern IC Manufacturing Cycle

- **Fabless** design now mainstream
  - IC designed in-house
  - Fabrication outsourced to foundry
- Externalities introduced
  - Fab: infrastructural, testing, calibration related additives
  - Design: third-party IP and tools, standard cells, models
- Multiple parties get involved
  - Difficult to guarantee returned IC genuinely matches original design

Does returned silicon reflect genuine design?
IC Trojan and Detection

- What is IC Trojan?
  - Malicious circuitry inserted on purpose by adversary
    - Not a bug or accidental modification
  - Inserted during design and fab steps
  - Dormant until triggered to get activated
    - Better catch while dormant to avoid consequences
    - Difficult to catch with small background power usage at dormant
      » Process variation can be larger
  - Consequences
    - Malfunction: performs incorrect operations, fails normal tasks
    - Breach of security and privacy: leaks sensitive/critical information

- Detecting Trojans via “power” or “current” side-channel measurement analysis
  - Want to detect any abnormal readings
  - Depends on circuit inputs that drive IC to lowest power states so extra leakage above expected deviation can be detected
Side-channel Approach

- Run sufficiently many test vectors for side-channel measurement
  - Increase chances to include **revealing** test vectors
- Use reference measurement values
  - Process-specific Trojan-free mean and deviation for all test vectors

\[ \mathbf{v}_N \ldots \mathbf{v}_2 \mathbf{v}_1 \rightarrow \text{Circuit Under Test (CUT)} \rightarrow \mathbf{x}_N \ldots \mathbf{x}_2 \mathbf{x}_1 \rightarrow \text{Compare} \rightarrow \text{Result} \]

\[ \mathbf{v}_N \ldots \mathbf{v}_2 \mathbf{v}_1 \rightarrow \text{Tool} \rightarrow \text{DB} \]

- Probability distribution DB of gate-level characteristics specific to fab process
- Done offline

\[ \mathbf{g}_1 \mathbf{g}_2 \ldots \mathbf{g}_N \rightarrow \text{N reference (golden) measurements} \]
Challenges of Side-channel Approach

- Trojan background power consumption too small
  - Noticeable only by *revealing test vectors*
- But how to find revealing test vectors?
  - No prior information
  - How many is sufficient?
- Chip I/O is bottleneck
  - Infeasible to export large number of measurements for off-chip analysis
- Intelligence of Trojan designer makes detection more difficult
  - Know vs. not-know the IC design
  - If knowledge enables to offset amount of Trojan power leakage, detection may be impossible
- Assuring detection reliability
  - How to reduce false positive and false negative rates?
Compressive Sensing as Solution

- Compressive sensing
  - Signal processing technique for recovering data with number of measurements proportional to sparsity of data (not size)
  - Uses simple encoding

- Why is compressive sensing applicable?
  - Revealing test vectors are sparse
  - Can reduce chip output requirement while capturing significant power leakage due to Trojans
DISTROY – Compressive Sensing Encoding

\[ x = [x_1 \ x_2 \ ... \ x_N]^T \] is buffered test vector output

DISTROY encoding: \[ y = \Phi \ x \]
- Compresses \( x \) (size \( N \)) in \( y \) (\( M \) RLCs) using \( \Phi_{MxN} \)
- \( M \ll N \)
- \( \Phi \): random measurement matrix
DISTROY – Compressive Sensing Decoding

Compressive sensing uses \textit{l1-norm minimization} decoding

- \( \mathbf{d} \) is sparse, thus recover \( \mathbf{d} = \mathbf{x} - \mathbf{g} \) directly
  - Of course, \( \mathbf{x} \) can be recovered from \( \mathbf{d} \)
- \( \mathbf{g} \) = corresponding expected output values for Trojan-free IC
- Process variation makes leakage current vary
  - \( \beta \): average leakage current contributed by Trojan gates
  - Small \( \beta \) makes detection more difficult \( \Rightarrow \) large overlap under curves
- Detection threshold \( \alpha \)
  - Tradeoff between false positive and negative rates: can optimize only one of them (not both)
  - Can we do better?
Enhance Detection with Testing Multiple Chips

- Group multiple chips by fab process
- To reduce false positives
  - Require all $P > 1$ chips meet detection criteria
- To reduce false negatives
  - Require at least $P$ out of $Q > P$ chips meet detection criteria
  - For fixed $P$, larger $Q$ yields fewer false negatives $\Rightarrow$ we can achieve both false positive and negative rates reasonably good.
Evaluation

- Benchmark circuit has 100 NAND gates
  - Built using ISCAS-85 c17
- Wrote logic simulation in C
  - Pre-ran all possible test vectors and cached results
- Trojan circuits
  - Placed 1 to 5 NAND gates at random locations
  - trojan-1/2/3/4/5
    - trojan-1 yields smallest leakage, thus most difficult to detect
- Metrics
  - Compression gain (N/M)
  - False positive rate
  - False negative rate
Expected Outcome

- Compressive sensing *advantage* $\implies$ achieves same margin of error with reduced number of measurements
  - Without compressive sensing: $N$ measurements needed
  - With compressive sensing: $N/k$ measurements should suffice
- Compressive sensing *tradeoff* $\implies$ reduced measurements for increase in false detection rates
  - How would false detection rates grow?
Detection Performance: Single Chip Testing

(a) Trojan size = 1 gate

(b) Trojan size = 5 gates

- About 4:1 to 5:1 compression gain (for false rates < 0.05)
  - Trojan size matters
- False rates go up quickly after reducing further from some $M$
Detection Performance: Multiple Chip Testing

- Testing multiple chips reduce both false rates
- We can address tradeoff with fixed P and adjustable Q
Summary

- DISTROY unconventional new way of using compressive sensing
  - Takes *test vector output* values as signal to compress
  - Substantially reduces chip output requirement related to detecting statistically rare events from large measurements

- Combined with testing multiple chips from same fab process, we can detect Trojans *reliably*
  - Despite inevitable tradeoff, we showed that both reasonably good *false positive* and *false negative* detection rates can be achieved

- We’re implementing DISTROY and plan to test against real chips with real Trojans
Extras
Consider 10-chip test example: $Q = 10$

- Fix $P$ first
  - $P = 2$ happens to meet required false positive rate

- Trojan-free IC (left curve)
  - Probability at least $P$ out of $Q$ (2 out of 10) chips power higher than $\alpha$ is very small $\implies$ false positive rate is small

- Trojan-containing IC (right curve)
  - Probability that any 9 of 10 chips all exhibit power lower than $\alpha$ is very small $\implies$ false negative rate is also small