SuperPUF: Integrating Heterogeneous Physically Unclonable Functions

Michael Wang, Andrew Yates, Igor L. Markov
University of Michigan, 2260 Hayward St., Ann Arbor, MI 48109

ABSTRACT

Physically Unclonable Functions (PUFs) combat counterfeit ICs by identifying each chip using inherent process variation. PUFs must produce sufficiently many bits, but replicating the same PUF design requires care since process variation and its spatial correlation may change in the next 10 years. Additional challenges arise in system-on-chip and heterogeneous 3D integration of diverse PUFs. Responding to these challenges, we introduce methods for combining PUFs, with provisions for sampling process variation throughout the IC. When multiple sources of entropy are available, our optimization algorithms select sources to maximize joint entropy and minimize physical overhead. Empirical validation uses SPICE simulations for a 45nm technology node.

1. INTRODUCTION

As entire industries grow more dependent on electronics, there is an increasing need for authentication of semiconductor devices and electronic systems. Counterfeit ICs are particularly challenging, as they are often cheaper than original ICs but less reliable; they can fail, slowly degrade, or undermine systems and networks. Counterfeit ICs damage IC suppliers' reputation and displace IC sales [6, 7].

To guard against counterfeit ICs and facilitate authentication, it is important to distinguish different chips produced from the same mask [17]. A straightforward approach to authentication is to store a secret key in non-volatile memory such as EEPROM. Cryptographic primitives such as digital signatures can then use the secret key to authenticate the IC [3]. Unfortunately, embedded non-volatile memory is not only expensive, but can also be probed, allowing the adversary to read secret keys and potentially clone authenticated chips [2, 1, 20]. As an alternative, researchers developed Physically Unclonable Function (PUF) designs that generate secret keys by sampling on-chip process variation — their introspective circuits produce different outputs on every chip and are sensitive to tampering [9, 10, 11, 13, 21]. The secret key is based on these small differences and is not stored explicitly — it is only available during PUF read-out. Nondestructive probing of a powered-up IC is much harder than probing a non-volatile memory on a powered-down IC. A reliable secret key facilitates a variety of cryptographic primitives. Incorporating a challenge-response protocol into the PUF design further complicates attacks [9, 15].

PUF readouts for a given chip must be reproducible over time and under different environmental conditions [4, 14]. On the other hand, they must provide sufficient differences across multiple chips and cover a broad range of possibilities, so as to defeat guessing. In this paper, we employ the concept of entropy to measure and compare the unpredictability of PUF readouts. We calculate the entropy of different PUF designs using statistical models of IC components and demonstrate algorithmic optimizations that maximize entropy. We believe that such techniques deserve a much wider use in PUF studies than previously reported.

Most PUF designs are local to a small section of an IC and compare multiple copies of one PUF circuit sensitive to process variation (Section 2). To produce more bits, one can use more copies, but this brings diminishing returns when the resulting bits are correlated in at least some process corners under at least some environmental conditions. Another looming problem is smaller uncorrelated process variation available to PUFs in circuits with FinFETs (where doping is reduced), depleted Si, and after the upcoming transition to 13.5nm-wavelength EUV-based manufacturing with directed self-assembly. To address these challenges, we advocate PUF designs that combine diverse sources of entropy placed throughout the IC. These sources may be sensitive to different types of variation and may be supplied by predesigned hard IP blocks. The idea to sample process variation that is sensitive to clock skew at selected spatially-distributed clock sinks has recently appeared in ClockPUF [22]. We generalize this idea to include other sources and reduce the added overhead.

Our contributions toward PUF theory and heterogeneous PUF integration are as follows:

- A physical architecture to link entropy sources through a single path and algorithms to optimize source selection to reduce implementation overhead (Figure 3).
- Combining different PUF circuits to produce a cohesive challenge-response system, e.g., from IP blocks.
- PUF considerations for 3D IC integration to take full advantage of relevant technologies.

The remainder of the paper is organized as follows. Section 2 provides an overview of known PUF designs and motivates our work on PUF integration. Section 3 shows the advantages of hybrid PUFs. Sections 4 describes process variation found on an IC and the methods used to calculate their entropy. The SuperPUF architecture and necessary support in terms of algorithmic EDA are covered in Sections 5-7. Section 8 reports empirical validation.

2. BACKGROUND

Threat models in IP privacy and overbuilding are reviewed in [17, Section 3]. PUF designs developed to counter such threats often use challenge-response pairs (CRPs) to authenticate an IC [9, 15]. The number of possible challenges should be large in order to increase the number of CRPs and to make the PUF difficult to mimic. The response bit is dependent on the challenge bits and the variation in the PUF circuitry. This variation is composed of non-dynamic process variation and dynamic variation which changes every time the PUF is used. For a PUF to be robust, the effects of dynamic variation should be minimized so that the CRPs do not change. Most PUFs cannot completely eliminate dynamic variation and require error-correcting codes [14].

2.1 State of the art in PUF design

Novel PUF designs are reported frequently.

Arbiter PUFs are composed of pairs of identical small delay paths, MUXes, and an arbiter. This type of PUF is discussed in [16] for use in FPGAs. The small delay paths are connected by MUXes to form two longer delay paths. The MUXes control how one small path pair connects to the next small path pair so that MUXes can vary the composition of the large paths. The small paths are usually composed of inverters and wires. A signal is sent through both long paths and an arbiter is used to detect which path propagates the signal through first. The challenge bits are used to control the MUXes and the arbiter produces the response bit. The main advantage of arbiter PUFs is that they have a large number of possible challenges (2^N for N MUXes). However, the generated responses are highly correlated which makes the responses more predictable.

Ring Oscillator (RO) PUFs generate CRPs by comparing the frequencies of identical ring oscillators. In [21], the frequency of a RO is measured by counting the number of oscillations that occur during a given number of clock cycles. The challenge bits are used to select which two ROs to compare, and the response bit is measured by comparing the oscillation counts of the ROs. The oscillation counts depend on the temperature, but the comparison between them does not. RO PUFs are good to use when there is little process variation because the multiple oscillations compound the small differences in the circuits into large differences in the output.

Clock PUFs use clock skews to generate CRPs. Introduced in [22], Clock PUFs select clock sinks and connect them to a central circuit where the skews are compared. The paths that carry the clock signals are designed to have the same delay so that ideally every signal arrives at the same time. Challenge bits use MUXes to select two sinks and an arbiter compares their delays to generate a response bit. Clock PUFs are connected to many parts of the IC which makes them difficult to separate and very sensitive to tampering

SRAM PUFs, unlike previously covered PUF designs, forego CRPs for a secret key by using the bistability of SRAM cells. On chip power-up, process variation biases an SRAM cell to start at either a 1 or a 0. A single cell can be used to generate one bit of a secret key, and a large secret key can be drawn from multiple cells. SRAM PUFs are simpler than most other PUF designs because they do not require precisely-timed circuits and SRAM arrays are widespread.

2.2 Statistical Models of Process Variation

We distinguish *manufacturing variations* from *slight physical differences between cloned PUF components*, such as small differences in route lengths that affect measured timing discrepancy. Unlike process variations, these differences are not unique to each chip and should not be relied upon to produce unique PUF responses. A third type is *dynamic variation*, including thermal fluctuations, EM noise and IR drop, as well as circuit-aging effects that occur over time, at different scales. To ensure stability, PUF responses should be as insensitive as possible to dynamic variation. On-chip temperature variation is problematic due to its impact on electrical parameters and its coupling to ambient temperature, which can be controlled when trying to sabotage PUFs. To lessen the impact of dynamic variation, most PUFs generate responses by comparing near-identical circuits, cancelling out the bulk of dynamic variation.

Statistical models of PUFs are built from component models, such as wires, transistors or small circuits (e.g., ring oscillators for RO PUFs). Component models must capture variance (σ^2) and correlation between the components (ρ). Most PUF designs ultimately use timing differences between sampling circuits to generate CRPs, calling for timing models of sampling circuits. Given that timing differences can be small, such models require SPICE accuracy.

Statistical models of sampling circuits should distinguish granular, spatial, and dynamic variation. Thanks to the Central Limit Theorem, normal distributions represent sums of random variables fairly accurately regardless of how individual variables are distributed — in practice, sampling circuits are affected by many sources of process variation. We represent the variance of granular variation by σ_g^2 and assume no correlation with other components: ρ_g=0. Spatial variation has variance σ_s^2 and exhibits correlation $\rho_s(r)$ at locations separated by distance r. These values can be derived using methods in [19], including the spherical equation to calculate ρ_g. In the definition below, ϕ represents the distance at which spatial correlation vanishes.

$$\rho_s(r) = \begin{cases} 1 - 3r/(2\phi) + r^3/(2\phi^3) & \text{if } r \le \phi \\ 0 & \text{otherwise} \end{cases} \qquad (2.1)$$

Empirical characterization of sampling circuits may be unavailable or prohibitively expensive. Instead, statistical models of the sampling circuits are developed based on gate models supplied with the standard cell library. Equation 2.2 calculates σ_g of a sampling N-component circuit with individual variances σ_{gi}. To estimate *spatial variance* σ_s in the case of nearby components, we assume full correlation, which results in arithmetic sum.

$$\sigma_g^2 = \sum_{i=1}^{N} \sigma_{gi}^2 \qquad \sigma_s = \sum_{i=1}^{N} \sigma_{si} \qquad (2.2)$$

By the Cauchy-Schwarz inequality, spatial variation results in a larger parameter variance in the final circuit than granular variation. While appropriate for simpler PUFs, Equations 2.2 would be insufficient for spatially distributed PUFs with complicated correlation structures, such as ClockPUFs.

3. THE NEED FOR SuperPUFs

Requirements for PUF readouts share some similarities with those for cryptographic random number generators [8] - they must contain sufficient entropy to defeat even partial prediction between multiple devices (e.g., by machine-learning attacks). On the same device, both cryptographic random numbers and PUF readouts must be insensitive to environmental conditions, such as ambient temperature. But PUF readouts from the same device must be repeatable. Our work seeks to increase the entropy available to PUFs, while ensuring stability of PUF readouts on individual devices. Whereas previous work usually treats entropy qualitatively, we propose quantitative metrics and optimizations.

Entropy as an Explicit Design Concern for PUFs

IC components experience process variation which can be separated into two types: *spatially-correlated* and granular. Spatial correlation decays with distance and vanishes at a sufficiently large scale. This classification is applicable to existing PUF constructs, which are typically confined to a small area of an IC and may experience spatial correlation if duplicated. When a PUF is designed to survive various environmental conditions and process corners, worst-case spatial correlations must be considered. The upcoming transition to manufacturing based on 13.5 nm EUV and directed self-assembly, as well as depleted silicon and FinFETs, promise to diminish process variation and limit the number of unique bits generated from cloned PUFs. To address these technology trends, we use Shannon's *diffusion principle* advocated in [18] and propose a distributed PUF design which spreads its circuitry across the IC to survive spatial correlation and draw additional entropy from interconnect delays. Depending on the variances of granular and spatial variation, heterogeneity can greatly increase total available entropy. Such PUFs are more difficult to sabotage, as they depend less on any one type of variation. IC designers need not bet on one PUF type, instead a SuperPUF selects an optimized portfolio of constituent PUFs best suited to a given manufacturing process and IC application.

Toward Greater Robustness

As a design principle, it is often assumed that every PUF, no matter how carefully designed, is susceptible to dynamic variation. Whereas PUFs convert continuous values (transition time, voltage, transistor strength, etc.) into bits by comparing those values to thresholds, minute dynamic variations of near-threshold values routinely produce undesirable fluctuations in generated bits. Since a small amount of fluctuation cannot be avoided, it is common to stabilize PUF output using error-correcting codes. The overhead of error-correction grows quickly with the amount of correctable errors, which depends on the impact and likelihood of dynamic variations. Two identically designed circuits must produce sufficiently different outputs sufficiently often to prevent dynamic variation from causing fluctuations in the response bit. To increase the number of possible responses, RO PUF and ClockPUF designs use numerous information sources with limited value ranges and compare them to each other.

PUFs for Heterogeneous IC Integration

Modern ICs are almost never designed from scratch, but rather combine previously-designed blocks, often from different vendors. 3D integration can stack several dice produced in drastically different manufacturing technologies. These new assemblies entered the consumer market in 2013, offering new levels of complexity and making them prime targets for counterfeiting. Developing PUFs for 3D IC authentication is important but challenging because such hybrid PUFs must accommodate diverse PUFs from individual dice. 3D ICs offer additional sources of entropy for PUFs, including process variation in through-silicon vias (TSVs) and die-to-die variation. The principles for designing SuperPUFs proposed in this work help analyze the impact of these additional sources. Our SuperPUFs can also mix the entropy generated by 2D and 3D sources to meet requirements and constraints of a specific IC design.

4. ENTROPY AND ROBUSTNESS OF PUFs

Accurate measurements of entropy based on the CRPs of a PUF are difficult to obtain for a PUF with a large number of unique responses. As shown in [8], an RNG based on a low-entropy source may pass standard tests for randomness but be open to exploitation. Poorly-designed PUFs may also exhibit much smaller entropy than intended. Tests based on future advances in machine-learning may help recognize low entropy and exploit it. Hence, we advocate statistical modeling based on process variation to estimate PUF entropy and compare the effectiveness of PUF designs.

Differential entropy h estimates the amount of information extractable from an N-dim random vector, which we assume (for convenience) to be normally distributed with the covariance matrix $\bar{\Sigma}$. In the PUF context, the vector represents contributions from individual entropy sources. Below, Δ is the accuracy (resolution) of time measurement.

$$h = 0.5\big[N \log_2(2\pi e) + \log_2 \det(\bar{\Sigma}/\Delta^2)\big] \qquad (4.1)$$

We propose using this information-theoretic metric for comparing PUFs based on identical or different designs.

Discrete entropy quantifies individual entropy sources and offers upper bounds on differential entropy. It can help quantify losses during integration and the extraction of discrete PUF readouts. To extract discrete entropy from a continuous distribution, one can perform quantization into Δ-sized sections, assign a single probability to each section, and apply Shannon's formula $H = \sum p_i \log_2 p_i$. Discrete entropy can be a more reliable metric of the final (sampled) distribution in an integrated PUF. However, when $\sigma \gg \Delta$, univariate differential entropy provides accurate estimates of discrete entropy. Depending on the accuracy of the measurement/sampling circuits and the amount of variance in entropy sources, this shortcut may or may not apply.

Robustness of PUFs discussed in Section 3 improves with coarser measurements, but this decreases extractable entropy. With process variation below measurement accuracy, no entropy is available. But when dynamic variation exceeds measurement accuracy, PUF readouts become unstable. Hence, it is more effective to use low-correlated high-entropy sources. *Robustness* can be estimated as the expected fraction of stable responses, calculated in terms of Δ and the amount of process variation in generating each response. Thus, one can capture tradeoffs with extracted *entropy* — greater entropy (say, with lower Δ) tends to reduce robustness. The discrete entropy of RO PUFs and some other designs, meets a ceiling with sufficient process variation, while PUF robustness tends to 1.0 in general.

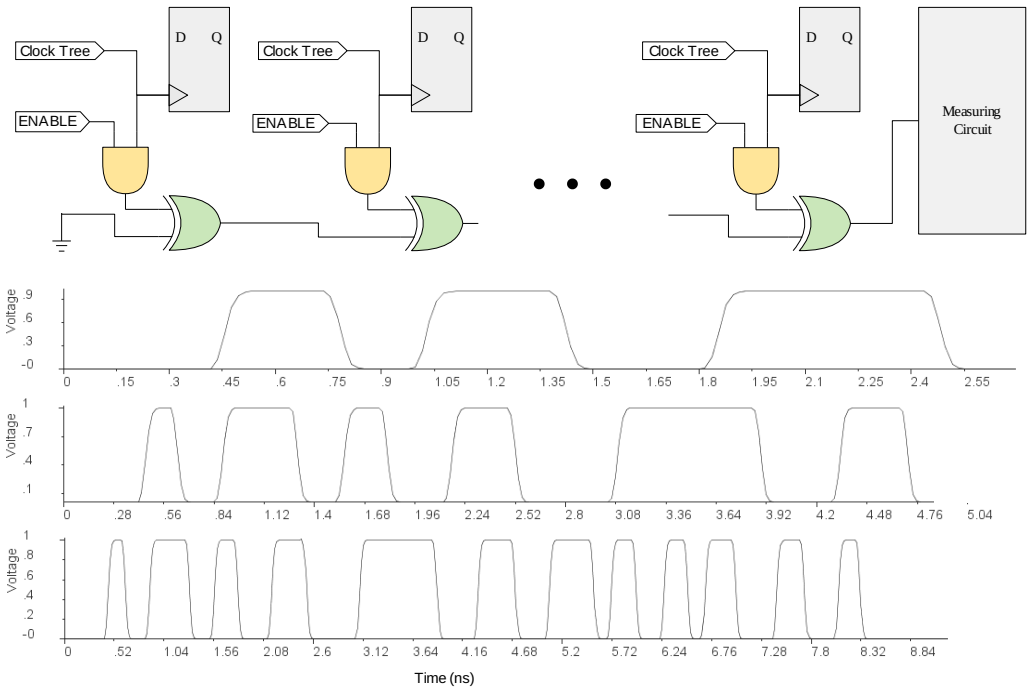

Figure 1: The rise and fall of SuperPUF: linking entropy sources (from ClockPUF) with a single path using `XOR` gates. SPICE waveforms are shown for 6 (top), 12 (middle) and 24 (bottom) `XOR` gates. To integrate hundreds of entropy sources, SuperPUF readout can be performed with a slower clock rate. Additionally, by scheduling `enable` signals connected to `AND` gates, it is possible to configure multicycle operation and integrate a large number of entropy sources, accounting for the delay along the path.

5. THE SuperPUF ARCHITECTURE

We now introduce our SuperPUF architecture using the reasoning from Sections 3 and 4 . Our illustrations draw on the ClockPUF infrastructure [22], which can be viewed as extracting entropy from spatially distributed sources. Section 6 describes algorithmic EDA support for SuperPUF, and Section 7 generalizes SuperPUFs further.

5.1 Combining On-chip Entropy Sources

We assume that on-chip entropy sources are represented by time-varying signals that depend on process variation, for example clock sinks in the ClockPUF [22]. Section 4 introduced ways to measure information content of such individual sources, as well as their joint information content. Multiple sources can be combined by designing an interconnect network. While the return network in ClockPUF provides a good example, we propose a different architecture for SuperPUF that is more flexible and efficient. Individual entropy sources are connected by a path, rather than a tree, and their time-varying signals are combined with the signal carried by the path using XOR gates, as illustrated in Figure 1. The choice of XOR gates is important — XOR gates do not possess logic don't-cares and propagate input changes to the output more faithfully than AND/OR gates. On the other hand, we use AND gates to handle (optional) enable signals that can be used to reduce power-consumption when PUF readout is not performed.

Figure 1 shows waveforms of signals collected on the common path in several configurations (SPICE simulations at the 45nm technology node). The impact of process variation is represented by the timing of rise and fall transitions in this waveform, and can be measured to produce PUF bits. A clear seperation between signal transitions is crucial for many reasons: (i) poor separation can complicate measurement, (ii) lack of separation may result in loss of information as positive and negative transitions will start cancelling out, (iii) lack of sharp waveforms may lead to chaotic behavior in measurement circuits and would thus require more reliable circuits with larger overhead, (iv) well-separated transitions make the information content of the Entropy Collection Path (ECP) independent on the ordering, which we use in algorithmic optimizations in Section 6.

Figure 2 describes measurement circuits used to collect PUF bits from paths introduced in Figure 1. Each path is forked, and delay buffers are inserted, alternated with RS latches that check which of their two inputs transitions earlier. Delay buffers are tuned to balance mean delays, remove design bias, and maximally expose the impact of process variations. Each latch measures one bit tied to a particular signal edge. The approach outlined so far has an apparent limitation — pulses that are too narrow will be filtered out by XOR gates with subsequent loss of information. Hence, we must lower-bound the time separation between adjacent rise and fall transitions, limiting the number of transitions that can occur within the clock cycle. This limitation is circumvented by disabling the inputs of XOR gates after one clock cycle, but allowing for *multicycle operation* of measurement circuits. One can also use a slower clock for SuperPUF readout to support more signal transitions per cycle.

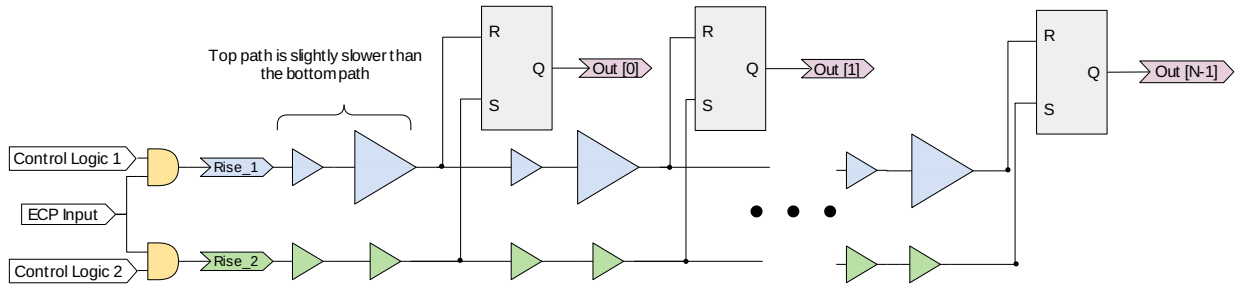

Figure 2: Our measurement circuit for extracting bits from the timing of signal transitions.

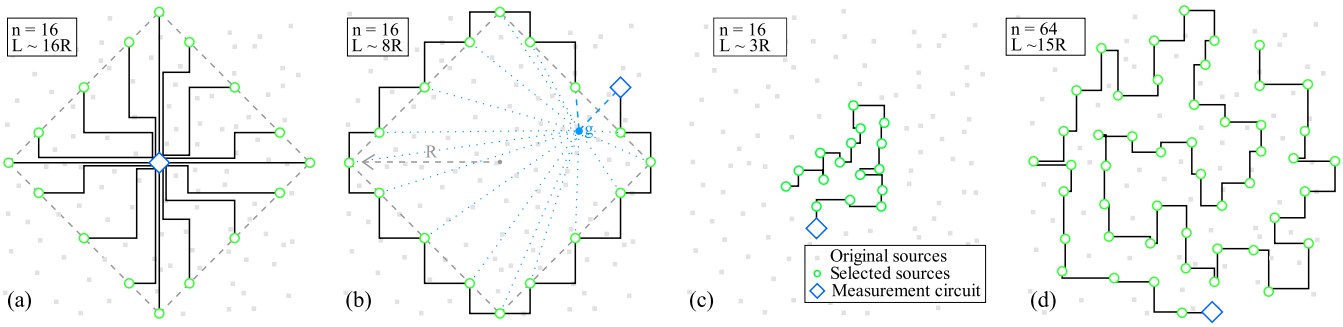

Figure 3: (a) entropy-source selection and routing in the original ClockPUF architecture [22], (b) SuperPUF routing for sources selected by the ClockPUF architecture — the ghost vertex and its zero-cost edges are shown in blue. Source selection and routing in the SuperPUF architecture are illustrated for ICs with (c) low and (d) high spatial correlation. Note that (d) taps four times as many sources as (a), but with a smaller wiring overhead. All four instances select entropy sources from the same initial pointset. L is the wiring overhead in terms of the radius R of the Manhattan diamond used in the ClockPUF architecture.

5.2 Reducing Wiring Overhead

Recall that ClockPUF [22] selects a subset of sinks of an on-chip clock tree, gently taps out arriving clock signals from sinks, and returns those signals to a central location for pairwise comparisons of transition times that are sensitive to process variation. Here entropy is drawn from clock skews, which are exceptionally robust to process variation and environmental conditions. The sinks are selected to be *approximately equidistant* from the central location (at distance $\sim R$),[1] and each sink is connected to a dedicated return path. With shortest paths, total wirelength is $L \approx nR$.

To reduce the wiring overhead of ClockPUF [22], SuperPUF *replaces the tree-like return structure with a single path* that collects entropy from selected sinks and feeds a measurement circuit that extracts PUF bits. When drawing entropy from clock sinks (to facilitate a comparison to ClockPUFs), SuperPUF links each selected sink to the path with an `XOR` gate to absorb skew information (Figure 1).

Approximately equidistant sinks in ClockPUF lie in a distinct thin diamond-shaped band (a Manhattan annulus) with extraction circuits in the center. As illustrated in Figure 3, the perimeter of the diamond is $8R$, regardless of n, and approximates a shortest path connecting all sinks. Given that $n \gg 8$ in practice, such a path is much shorter than the ClockPUF return network. The equidistance condition is now unnecessary, which allows us to use a greater number of clock sinks, not limited to the diamond band. However, finding such a path requires at least as much computational effort as solving the planar Travelling Salesman Problem (TSP). On the other hand, the freedom of sink selection helps reduce wiring overhead by using closely located sinks. It also allows us to account for correlations between sinks.

The proposed physical architecture for path-based SuperPUF leads to the following optimization problems

- selecting entropy sources, and
- connecting them by an entropy-collection path (ECP) to increase joint entropy with low overhead.

We develop two algorithmic techniques: (i) a combination of Travelling Salesman Problem (TSP) heuristics with Reverse Cuthill-McKee (RCM) matrix ordering from numerical analysis, and (ii) a multiobjective greedy approach.

6. DESIGN AUTOMATION FOR SuperPUFs

To support the implementation of SuperPUFs and decrease their overhead, we developed several algorithmic optimizations, outlined in Figure 4 for the simple case when all potential entropy sources have *a priori* locations. In the context of ClockPUF infrastructure, Steps 1-3 are performed as follows: potential entropy sources are clock sinks, the entropy matrix is constructed based on accurate circuit simulations that account for process variation, and the distance matrix stores Manhattan distances between sink coordinates. The remainder of this sections deals with Steps 5 a-c from Figure 4, whereas complications arising from heterogenous sources and 3D IC integration are discussed in Section 7.

[1] R is the smallest radius that facilitates the required number of equidistant locations [22].

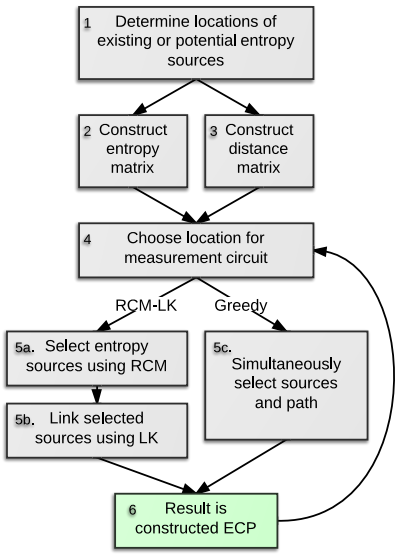

Figure 4: The process of creating an ECP. For Step 1: in ClockPUF, the eligible entropy sources are all clock sinks; in SuperPUF eligibile entropy sources span all PUFs which could potentially be placed on the IC. Steps 4-6 can be re-tried to improve results.

6.1 Optimizing the Entropy Collection Path

For the selection of entropy sources, we noted that the amount of entropy collected by the path is largely independent of the sink order. Then minimizing the wiring overhead reduces to the problem of finding a least-cost path through selected vertices, which is related to the Travelling Salesman Problem (TSP) through the use of a *ghost* vertex (Figure 3b).

Observation 1. View n points in the Manhattan plane as vertices of a clique G with Manhattan distances as edge weights. Form a larger clique G' by adding a *ghost vertex* g connected to all other vertices in G through 0-cost edges. Then each least-cost *path* of G corresponds (one-to-one) to a least-cost *tour* in G' with the two edges adjacent to g removed.

Observation 2. Without a ghost vertex, removing a longest edge in a least-cost cycle can produce a suboptimal path.

We implemented the *Lin-Kernighan* (LK) TSP heuristic [12] with random initial solutions and adapted it to produce low-cost paths by removing the ghost vertex and its adjacent edges from a low-cost cycle. LK-TSP runs in $O(n^{2.2})$ time, is fast in practice, and finds solutions within 1-2% of optimal (better with several independent starts). Optimal TSP solvers, e.g., *Concorde*, can also be used and scale to fairly large TSP instances, but take much longer than heuristics.

6.2 Ordering and Selecting Entropy Sources

The use of TSP heuristics and the ghost-vertex method allows one to link up a selection of sinks in a way that minimizes path length. However, it does not address the sink-selection problem. Our technique for sink selection draws inspiration from the *reverse Cuthill-McKee algorithm* from numerical analysis which speeds up sparse-matrix linear algebra [5]. Given a symmetric $n \times n$ matrix (considered sparse), Cuthill-McKee produces an ordering which renumbers the n rows to reduce *matrix bandwidth*, i.e., it views the matrix as a graph and seeks to cluster closely connected vertices. Reverse Cuthill-McKee traverses that ordering from the end to beginning. In our application, *ordering closely correlated vertices next to each other allows us to select every K-th vertex to improve diversity.* In other words, at most one vertex should be selected from each cluster of correlated vertices.

Algorithmically, Cuthill-McKee is a variant of Breadth First Search (BFS). It operates on a binary $n \times n$ matrix which may be interpreted as an adjacency matrix of a graph. In the context of SuperPUF, given n entropy sources, we consider an $n \times n$ matrix of *conditional entropies* between pairs of sinks. Uncorrelated pairs are more desirable. Such pairs are represented by higher numerical entries. This $n \times n$ matrix may be interpreted as an adjacency matrix of a weighted graph. While the matrix is not sparse, the intuition behind the RCM method still applies. Algorithmically, we replace the BFS traversal by a Dijkstra traversal which can account for edge weights (the $\log n$ increase in algorithmic complexity is negligible in our application). To select m sinks from a resulting ordering, we select every $\frac{n}{m}$th sink.

6.3 Greedy Optimization with Lookahead

While LK-TSP reliably produces near-optimal solutions, it only addresses half the problem as it does not account for entropy. Combining LK-TSP with the RCM-based heuristic accounts for both distances and entropy, but such a two-step optimization leaves room for improvement. Hence, we develop a simultaneous optimization. Starting at the sink closest to PUF readout, we iteratively select the next entropy source to minimize wiring overhead, subject to meeting preset thresholds for conditional entropy and distance.[2] Avoiding highly-correlated sources improves robustness of readout and helps collect sufficient entropy with smaller overhead. Avoiding sinks that are too close helps ensure a "clean" waveform (Figure 1). This algorithm may stop in a deadend when greedy selection exhausts eligible vertices.We then backtrack and employ a *lookahead* for the number of eligible vertices after each possible path continuation, as seen in next-sink selection pseudocode below.

Function: `choose_next_sink`
Input: Sink set $S[1...n]$, matrix $C[1...n][1...n]$ of entropy values, eligibility bitvector $E[1...n]$, path $P[1...i]$, int cutoff, int sinksNeeded
Output: path $P[1...i+1]$

```
1. if(P.size() == sinksNeeded) return true
2. if(numEligible(E) == 0) return false
3. candidates[1...5] = find_candidates(S,E,cutoff,P.back())
4. for (i = 1 to 5) {
5.     P.add(candidates[i])
6.     E' = update_eligibility(S,C,cutoff,candidates[i])
7.     if (choose_next_sink(S,C,E',P,sinksNeeded))
8.         return true
9.     P.pop(candidates[i])
10. } return false
```

Since the results produced by this fast algorithm depend on the starting location (PUF readout), one can run it from several starting locations and choose the best result.

[2]Conditional entropy is defined using conditional probabilities in the Shannon formula. In our context, conditioning is performed based on previously selected entropy sources.

7. MORE GENERAL SuperPUFs

The SuperPUF methodology and algorithms can be used with multiple spatially-distributed entropy sources, including RO PUFs, but this introduces complications. **First**, PUFs that require different readout regimes (ClockPUF, RO PUF, etc) must be multiplexed on a single ECP with appropriate `enable` signals, or connected by disjoint ECPs. **Second**, the introduction of entropy sources with flexible locations (such as RO PUFs) requires revisions in Steps 1-3 of Figure 4. A simple solution is to consider a large number of possible RO PUF locations and estimate their correlations as functions of distance. Only a small subset will be chosen by our algorithms. A more directed optimization that places entropy sources along an ECP is also possible. **Third,** estimating conditional-entropies[3] between heterogenous sources, such as small inverters and isolated PUFs, may require more complicated simulations. Once these issues are resolved for a particular SuperPUF configuration, algorithms in Section 5 select entropy sources and route connections between them.

In the context of 3D IC integration, through-silicon vias (TSVs) act as additional entropy sources that can be linked by a path. TSVs must also be accounted for as interconnects in both the entropy matrix (Figure 4 Step 2) and the distance matrix used by the path selection heuristics (Figure 4 Step 3). Pairwise distances between entropy sources on adjacent 2D dice are calculated by first finding the TSV that ensures the shortest total distance (the sum of the Manhattan distances on individual 2D dice) and then adding the cost of the TSV to that of the planar interconnects. As explained earlier, it is possible to consider a large number of possible TSV locations and use our algorithms to include only some of them in the ECP. TSVs are particularly useful when they link *uncorrelated* entropy sources on adjacent 2D dice, reducing interconnect overhead of PUF integration.

8. EMPIRICAL VALIDATION

We implemented the proposed algorithms in C++ and compiled them with g++4.7 on a Linux system. CMOS parameters represent a 45nm technology. Accurate circuit simulation was performed by 500x Monte Carlo runs of ngSPICE-25 to model process variation. The performance of SuperPUF on standard tests (robustness, uniqueness, randomness, etc) is inherited from the underlying PUF components. Here we we only demonstrate salient features of SuperPUFs.

Path construction

To compare the technique outlined in Sections 6.1 and 6.2 to greedy optimization from Section 6.3, we benchmarked them using clock networks from the ISPD 2010 Clock-network Synthesis Contest organized by Intel and IBM. For each benchmark, we generate a path using each technique.[4] The RCM-LK paths always connect to 64 sinks. The greedy approach *targets* 64 sinks but often stops short due to eligibility constraints. Results in Table 1 indicate that despite reasonable derivable *total discrete entropy*, RCM-LK paths tend to be rather long. We evaluate the efficiency of a path by calculating its discrete entropy/mm. This metric favors the greedy approach, which usually selects adjacent sinks that are close to each other. RCM can be biased toward closer sinks by dividing each entry in the entropy matrix by the distance between respective two sinks, but this does not significantly improve entropy/mm.

[3]We experimented with *differential* and *discrete* calculations for pairwise conditional entropy, and Pierson correlation. With no clear winner, we chose discrete conditional entropy.

[4]Our LK-TSP implementation has been extensively benchmarked against a much slower optimal solver and shown to produce solutions within 1-2% from optimal solutions.

Table 1: Path overhead of the greedy approach vs RCM-LK. Discrete entropy is expressed in bits. Wire length is expressed in millimeters.

Bench marks	Sinks	Greedy		RCM-LK	
		Discrete entropy	Wire length	Discrete entropy	Wire length
01	1107	101.25	41.09	75.23	53.86
02	2249	97.72	38.17	89.35	70.58
03	1200	43.41	7.13	36.96	11.11
04	1845	62.30	13.98	42.89	14.57
05	1016	39.74	7.01	36.01	13.51
06	981	37.99	6.08	30.96	9.01
07	1915	63.12	11.56	45.93	12.94
08	1134	43.86	9.21	33.89	10.57

In summary, near-optimal TSP solvers do not help obtain best results, as RCM is relatively weak in sink selection. Our multiobjective greedy approach with lookahead finds better paths. Other path-selection strategies, including *simulated annealing*, were inferior in our experiments.

Dependencies on spatial correlation

Using Equation 2.1, Figure 5 illustrates how changes in spatial correlation and the distance between sources affect the entropy and robustness of an RO-based SuperPUF. Note that both the number of extractable bits and % stable bits plateau as distance approaches ϕ, the point at which spatial correlation vanishes.

9. CONCLUSIONS AND PERSPECTIVES

Our work draws motivation from several trends and challenges. One is the decrease of process variation with upcoming major changes in IC manufacturing, which may undermine existing PUF constructs. Some of these concerns, along with the need to disperse the sources of randomness on the chip, have been reviewed in [18]. Another major development is the integration of multiple PUFs from reusable IP blocks and new entropy sources, increasingly likely with 3D ICs. Viewing these challengees as opportunities, we develop a technique for integrating distributed on-chip entropy sources, termed SuperPUF, that dramatically reduces wiring overhead compared to recently published related techniques. Our empirical validation shows how SuperPUFs adapt to different process-variation profiles.

In the context of our empirical validation, we would like to particularly emphasize the differences between simulation-based validation and that based on test chips. Unlike most prior work, our empirical methodology includes entropy calculations. However, evaluating PUF entropy from test chips requires an unacceptably large number of defect-free ICs. Equally important is our ability to experiment with a range of process-variation profiles — this ability is not available with manufacturing-based validation. Since test-chip validation often uses full-chip synthesis, we note that proposed circuitry is fairly independent from the main netlist, and

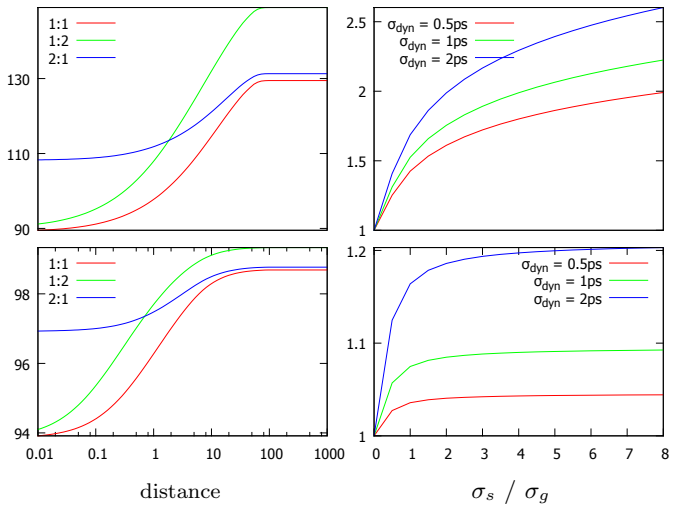

Figure 5: The impact of spatial variation on an RO-based SuperPUF. [Left:] we vary the distance between constituent RO PUFs and observe changes in extractable differential entropy [upper-left] and robustness (% stable bits) [lower-left], for three different ratios of variances of spatial vs granular variation (σ_s^2/σ_g^2). [Right:] we plot multiplicative gains in entropy [upper-right] and robustness [lower-right] against varying spatial versus granular variation ratios. The three lines represent three tested levels of dynamic performence variation: low, medium and high.

PUF readout can be performed in dedicated mode. Hence, full-chip synthesis should not affect relevant parameters.

We have not explored in this paper the construction of strong PUFs, but this can be done in several ways, for example through the use of the AES *counter* mode. A more economical approach is to use programmable delay buffers. Here, the challenge is to ensure that the amount of entropy in PUF response for each challenge is approximately the same.

Our research has outlined extensions of SuperPUFs which can accomodate multiple types of on-chip entropy sources. Such heterogenous SuperPUFs are particularly pertinent during system-level design when many available IP blocks must be gainfully combined. These considerations are amplified in the context of 3D IC integration which can combine 2D dice optimized for random logic, FPGAs, SRAM, analog, etc — each die provides a unique, uncorrelated entropy source, which we connect by an ECP. We hope that this work boosts the development of high-quality, licensed and trusted commercial PUF IP, especially given pending technological developments that promise to lower granular process variation — 13.5nm-wavelength EUV-based manufacturing with directed self-assembly, as well as depleted Si FinFETs.

Acknowledgment. This work was partially supported by the NSF Award 1162087.

10. REFERENCES

[1] R. Anderson, M. Kuhn. "Low-Cost Attacks on Tamper Resistant Devices", *IWSP: Int'l Workshop Sec'ty Protocols*, 1997.

[2] R. Anderson, M. Kuhn, "Tamper Resistance - a Cautionary Note", *2nd USENIX Workshop Elec. Commerce*, Nov 1996.

[3] F. Armknecht, R. Maes, A.R. Sadeghi, B. Sunar, P. Tuyls, "Memory Leakage-Resilient Encryption Based on Physically Unclonable Functions", *Advances in Cryptology* (ASIACRYPT), *LNCS* 5912: 685–702, 2009.

[4] F. Armknecht, R. Maes, A.R. Sadeghi, F.X. Standaert, C. Wachsmann, "A formal Foundation for the Security Fatures of Physical Functions", *IEEE Symposium on Security and Privacy (SSP)*: 397-412, May 2011.

[5] E. Cuthill, J. McKee,"Reducing the Bandwidth of Sparse Symmetric Matrices", *24th Nat'l Conf. ACM* 1969:157-172.

[6] Defense Science Board (DSB) Study on High Performance Microchip Supply, 2005.

[7] Defense Industrial Base Assessment: Counterfeit Electronics Study by U.S. Dept. of Commerce Bureau of Industry & Security Office Of Tech. Evaluation, 2010.

[8] D. E. Eastlake, J. I. Schiller, S. Crocker, "Randomness Requirements for Security", *BCP 106, RFC 4086*, 2005.

[9] B. Gassend, D. Clarke, M. van Dijk, S. Devadas, "Controlled Physical Random Functions", *ACSAC* 2002: 149-160.

[10] J. Guajardo, S.S. Kumar, G. Schrijen, P. Tuyls, "FPGA intrinsic PUFs and their use for IP protection", *Crypto Hardware & Emb Sys (CHES)*: 63-80, 2007.

[11] R. Helinski, D. Acharyya, J. Plusquellic, "A Physical Unclonable Function Defined Using Power Distribution System Equivalent Resistance Variations", *DAC*: 676-681, 2009.

[12] K. Helsgaun, "An Effective Implementation of the Lin-Kernighan Traveling Salesman Heuristic," *European Journal of Operational Research* 126(1): 106-130 (2000).

[13] D.E.Holcomb, W.Burleson, K.Fu, "Power-up SRAM State as an Identifying Fingerprint and Source of True Random Numbers",*IEEE Trans. Comp.* 58(9):1198–1210, Sep 2009.

[14] S. Katzenbeisser, Ü. Kocabas, V. Rožić, A. Sadeghi, I. Verbauwhede and C. Wachsmann, "PUFs: Myth, fact or busted? A Security Evaluation of Physically Unclonable Functions (PUFs) Cast in Silicon", *CHES* 2012: 283-301.

[15] J.W. Lee, A. Lim, B. Gassend, G.E. Suh, M. van Dijk, S. Devadas. "A Technique to Build a Secret Key in Integrated Circuits for Identification and Authentication Applications", *Symp. VLSI* 2004: 176-179.

[16] M. Majzoobi, F. Koushanfar and S. Devadas. "FPGA PUF Using Programmable Delay Lines", *WIFS* 2010: 1-6.

[17] M. Rostami, F. Koushanfar, J. Rajendran, R. Karri, "Hardware Security: Threat Models and Metrics", *ICCAD* 2013.

[18] M. Rostami, J. B. Wendt, M. Potkonjak, F. Koushanfar, "Quo Vadis, PUF?" *DATE 2014*.

[19] S. R. Sarangi, R.Teodorescu B. Greskamp, F. Koushanfar, J. Nakano, A. Tiwari, and J. Torrellas, "Varius: A model of process variation and resulting timing errors for microarchitects", *IEEE Trans. Semicon. Manuf.* 21(1):3-13, 2008.

[20] S. P. Skorobogatov, "Semi-invasive Attacks - a New Approach to Hardware Security Analysis", *Tech. Rep.* UCAM-CL-TR-630. *Univ. Cambridge Comp. Lab.*, April 2005.

[21] G.E. Suh and S. Devadas, "PUFs for Device Authentication & Secret Key Generation", *DAC* 2007: 9-14.

[22] Y. Yao, M.-B. Kim, I.L. Markov, F. Koushanfar, "ClockPUF: Physical Unclonable Functions Based on Clock Networks", *DATE* 2013:422-427.