Fall 2015

On the deployment of on-chip noise sensors

Tao Wang

Department: Electrical and Computer Engineering

ON THE DEPLOYMENT OF ON-CHIP NOISE SENSORS

by

TAO WANG

A DISSERTATION

Presented to the Faculty of the Graduate School of the

MISSOURI UNIVERSITY OF SCIENCE AND TECHNOLOGY

In Partial Fulfillment of the Requirements for the Degree

DOCTOR OF PHILOSOPHY

in

COMPUTER ENGINEERING

2015

Approved by

Dr. Yiyu Shi, Advisor
Dr. Daryl Beetner
Dr. Jun Fan
Dr. Minsu Choi
Dr. Maggie Cheng

ABSTRACT

The relentless technology scaling has led to significantly reduced noise margin and complicated functionalities. As such, design time techniques per se are less likely to ensure power integrity, resulting in runtime voltage emergencies. To alleviate the issue, recently several works have shed light on the possibilities of dynamic noise management systems. Most of these works rely on on-chip noise sensors to accurately capture voltage emergencies. However, they all assume that the placement of the sensors is given. It remains an open problem in the literature how to optimally place a given number of noise sensors for best voltage emergency detection. In the first chapter, the problem of noise sensor placement is defined along with a novel sensing quality metric (SQM) to be maximized.

The threshold voltage for noise sensors to report emergencies serves as a critical tuning knob between the system failure rate and false alarms. In the second chapter, the problem of minimizing the system alarm rate subject to a given system failure rate constraint is first formulated. It is further shown that with the help of I_{ddq} measurements during testing which reveal process variation information, it is possible and efficient to compute a per-chip optimal threshold voltage threshold.

In the third chapter, a novel framework to predict the resonance frequency using existing on-chip noise sensors, based on the theory of 1-bit compressed sensing is proposed. The proposed framework can help to achieve the resonance frequency of individual chips so as to effectively avoid resonance noise at runtime.

ACKNOWLEDGMENTS

I would like to thank my PhD advisor, Professor Yiyu Shi, for supporting me during these past five years. Prof. Shi is the funniest advisor and one of the smartest people I know. I hope that I could be as lively, enthusiastic, and energetic as Prof. Shi. Prof. Shi has been supportive and has given me the freedom to pursue various projects without objection. He has also provided insightful discussions about the research. I also have to thank the members of my PhD committee, Professors Daryl Beetner, Jun Fan, Minsu Choi and Maggie Cheng for their helpful career advice and suggestions in general.

I thank all the members in Prof. Shi's group, you gave me lots of help during my PhD. I am so happy to have you in my life. I also thank my friends in Rolla for providing support and friendship that I needed.

I especially thank my family. My hard-working parents have sacrificed their lives for me and provided unconditional love and care. I love them so much, and I would not have made it this far without them. Special thanks to the best gifts in my life, my two daughters, who filled my life up with love and warmth.

TABLE OF CONTENTS

Page

SECTION

LIST OF ILLUSTRATIONS

Figure Page

LIST OF TABLES

Table Page

1. EAGLE-EYE: A NEAR-OPTIMAL STATISTICAL FRAMEWORK FOR NOISE SENSOR PLACEMENT

1.1. INTRODUCTION

The continuous increase in power density brought by the CMOS scaling has resulted in a monotonic decrease of the supply voltage. On the other hand, to avoid excessive leakage power, the threshold voltage cannot be scaled at the same pace, as illustrated in Figure 1.1 [1]. The reduced noise margin, along with the boost in functional complexity, has posed severe threat to the power integrity of the chips. Noise margin violation, which is also known as voltage emergency, leads to undesired effects such as delay degradation, timing violation, etc. which may finally cause system malfunctioning [11, 33].

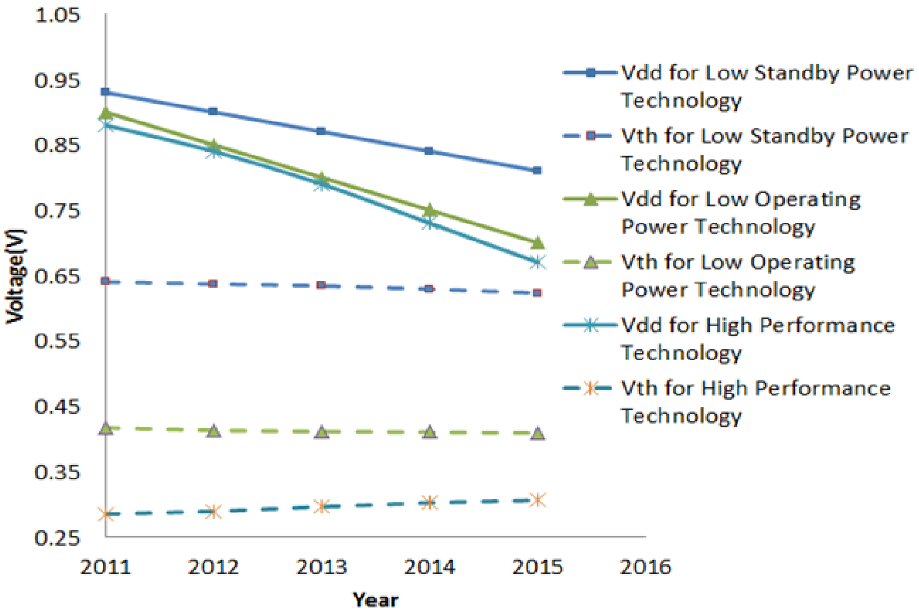


Figure 1.1: Supply voltage and threshold voltage scaling trends in International Technology Roadmap for Semiconductors[1]. By 2015 the voltage margin for Low Standby Power (LSP) technology will drop below 0.2 V

The strict definition of voltage emergency is the situation when the amplitude of the noise exceeds a given threshold voltage Vt for a minimum duration (threshold time) of T_t [24]. In other words, the definition captures not just the amplitude of noise, but also the temporal span. An example is shown in Figure 1.2, the duration of the first peak is longer than the threshold time to cause voltage emergency. On the other hand, the second peak is too short to cause it. The rationale behind this is that the impact of power supply noise on timing is more of an accumulative effect, and a very narrow voltage droop is unlikely to cause any timing issues. Conventionally, on-chip power integrity is ensured through design time approaches such as decap insertion [5, 35] or metal wire sizing [20] [15]. However, increasingly stringent design constraints with narrowed noise margins and complicated functionalities make such practices expensive, if not impossible, to address all power integrity issues at design time. As such, a recent trend of research innovation [11, 16, 27, 33] is to employ runtime noise management systems at the microarchitecture level to address the power integrity issue. Most of them adopted a centralized control system with distributed noise sensors, and were based on the detection of voltage emergencies. Obviously, the quality of such a microarchitecture level solution with runtime noise management greatly depends on the locations of those distributed noise sensors. Such a problem is, however, largely not addressed in literature.

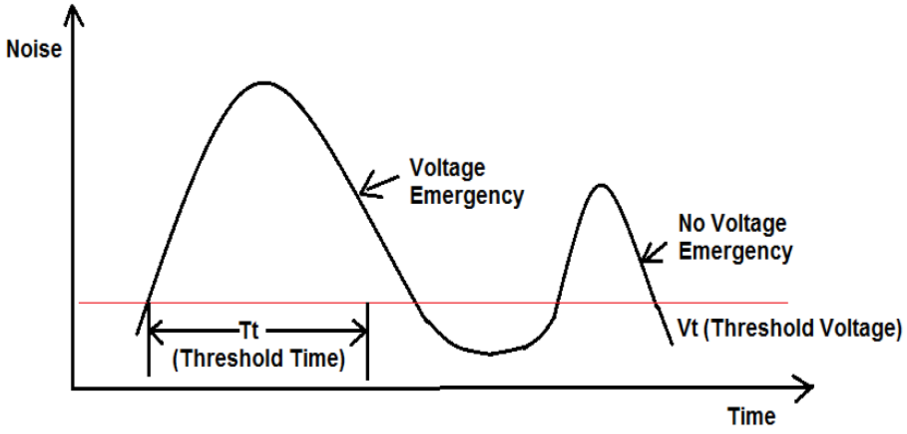


Figure 1.2: Definition of voltage emergency

How to place a limited number of noise sensors is important in order to minimize the hardware overhead and, at the same time, minimize the miss rate of voltage emergency detections, where the miss rate is defined as the probability of noise sensors not detecting any voltage emergencies, while a voltage emergency in fact occurring somewhere on the chip. Failure to detect voltage emergencies may lead to severe performance penalties. For example, the study in [11] showed that undetected voltage emergencies can induce as much as 17% delay degradation. The runtime noise management systems and the associated noise sensor placement problem will become increasingly important in the near future with even tighter noise margin and higher computing demand. Hence a rigorous study in this area is warranted.

In this chapter, the noise sensor placement problem for microarchitecture-level runtime noise management is formally formulated, and a statistical near-optimal framework is devised to solve it while considering the correlated noise distributions. The main contents of this chapter are as follows: 1) A formal formulation of the noise sensor placement problem for micro-architecture level runtime noise management; 2) A novel metric called sensing quality metric (SQM) to quantitatively evaluate the quality of any noise sensor placement; 3) An efficient approximation algorithm with O(sn) complexity, where n is the number of candidate nodes for sensor placement and s is the number of sensors placed; 4) A proof that the proposed algorithm can achieve the best-possible approximation to the optimal solution among all polynomial-complexity algorithms.

The remainder of this chapter is organized as follows. Related background information is reviewed in Section 1.2, and the noise sensor placement problem is formulated in Section 1.3. Proposed algorithm is presented in Section 1.4. Concluding remarks are given in Section 1.6.

1.2. PRELIMINARIES

1.2.1. Microarchitecture-level Runtime Noise Management. Most existing microarchitecture-level runtime noise management systems adopt a structure with distributed noise sensors and a centralized control system where the voltage emergencies are determined by comparing the sensed noise with a threshold. There are two different types of thresholds. Following the same conventions as [11], in this chapter the hard threshold as the strict constraint beyond which the system will malfunction (e.g. timing violation) are defined. On the other hand, the soft threshold is less strict, beyond which the system becomes less reliable due to effects such as delay degradation. Noise emergency is defined as the time when the noise surpasses/violates either the soft or the hard threshold, depending on the application. Depending on the way voltage emergencies are handled, microarchitecture-level runtime noise management can be divided into three different categories: retroactive system with soft threshold [16, 27], retroactive system with hard threshold [33], and proactive system with soft threshold [11].

Retroactive systems with soft threshold voltage use noise sensors to monitor the supply voltage for specific soft threshold crossings, which indicates potential system reliability reduction. When this occurs, the instruction execution will then be throttled at the microarchitecture-level to prevent potential system failures. However, significant overhead is induced by over-protecting the systems to ensure correctness. Retroactive systems with hard threshold improve such scheme by allowing the errors to occur, but system states need to be rolled back to restore the correct values once the noise emergency is detected. Proactive systems with soft threshold, on the other hand, try to recognize and track the patterns of activities that may lead to voltage emergencies, and hence invoke the throttling mechanism based on the prediction to prevent voltage emergencies from actually happening.

In either of these approaches, the optimal deployment of noise sensors (e.g., sensor design, placement, etc.) plays a central role to accurately capture voltage emergencies with lowest overhead.

1.2.2. Noise Sensor Placement. For dynamic noise management such as throttling (e.g.[11, 33]), it is only needed to know whether noise emergencies have occurred rather than the detailed noise map, which requires significantly less information. Consider a motivational design with only one single (and fixed) noise hot spot that is uncorrelated to other parts on the chip. In this case, only one noise sensor right at the hot spot is sufficient to detect all emergencies.

For noise sensor placement, the goal is to provide a binary decision on whether a voltage emergency has occurred somewhere on chip. The goal of the noise sensor placement is to accurately monitor and report the voltage emergency (where the chip max noise larger than a given threshold t) with a limited number of noise sensors and the binary information provided by the noise sensors.

1.3. PROBLEM FORMULATION

In this section, it is formally formulated the noise sensor placement problem, and the target metric to be optimized.

1.3.1. Problem Statement. It is assumed that the following information is given as input: 1) n candidate nodes in the power grid for noise sensor placement;2) threshold voltage t for the voltage emergencies, which is specified by the designer and is the same for all the sensors; 3) the total number of noise sensors t to be placed.

The objective is to identify s nodes $r_i (1 \leq i \leq s)$ out of the n candidates to be sensed, so that the miss rate of the voltage emergency detection is minimized. The candidate nodes are those which allow noise sensor placement. Again, the miss rate is defined as the probability that the placed noise sensors do not detect any voltage emergencies while a voltage emergency does occur somewhere on chip, including those places that are not allowed to place a noise sensor. Note that the actual locations of the sensors can be anywhere in the area adjacent to r_i. However, for the simplicity of presentation, in this chapter it is said a sensor is being placed at r_i if the sensor is used to sense the voltage at r_i.

In the formulation, one challenging problem is to quantitatively evaluate the miss rate with given noise sensor placements. As it is impossible to obtain all possible voltage drop (noise) waveforms through transient simulation, same as [37] it is proposed to model the noise ΔV_i , including the fluctuations in the reference voltage, noise sensor process variation, etc., at any node i of the power grid as a random variable. Those random variables have different means and variances, and are correlated spatially. Specifically, the noise of i-th node, either Gaussian or non-Gaussian, can be represented as [37].

$$\Delta V_i = F_i(\Delta X) = H_i(G) + \Delta R_i \tag{1.1}$$

where ΔX is a set of common correlated factors that result the variation of voltage noise through function F_i. Through modelling techniques, the noise can be represented by function $H_i(G)$, where G is an m-dimensional uncorrelated random variable that models the global variation sources (common for all nodes) which can be extracted from ΔX through either principle component analysis (PCA) for Gaussian or independent component analysis (ICA) for non-Gaussian distributions of ΔX. The dimension of m decides the approximation accuracy. In addition, ΔR_i models the independent source of noise variation specific to node i which comes from model error, noise sensor process variation, fluctuations in reference voltage. In addition, the functional forms of F_i and H_i can be either linear or nonlinear [7]. In the context of such statistical formulation, it is ready to put forward a novel sensing quality metric that can be computed without involving Monte Carlo simulations.

1.3.2. Sensing Quality Metric (SQM). Mathematically, the miss rate can be cast as

$$\text{Miss Rate} = P(\Delta V_{max} \leq t|\max(\Delta V_{r_i}(1 \leq i \leq s)) \tag{1.2}$$

where ΔV_{max} is the maximum noise among all the nodes in the power grid, $\Delta V_{r_i}(1 \leq i \leq s)$ are the noise at the s nodes where the sensors are connected, and t is the threshold voltage.

Equation (1.2) still provides little insight into the optimization problem. It can further simplified by using the Bayes law as follows.

$$\begin{aligned}
&P(\Delta V_{max} \geq t | \max(\Delta V_{r_i}(1 \leq i \leq s)) \leq t) \\
=&1 - P(\Delta V_{max} \leq t | \max(\Delta V_{r_i}(1 \leq i \leq s)) \leq t) \\
=&1 - \frac{P(\Delta V_{max} \leq t, \max(\Delta V_{r_i}(1 \leq i \leq s)) \leq t)}{P(\max(\Delta V_{r_i}(1 \leq i \leq s)) \leq t)} \\
=&1 - \frac{P(\Delta V_{max} \leq t)}{P(\max(\Delta V_{r_i}(1 \leq i \leq s)) \leq t)}
\end{aligned} \tag{1.3}$$

where the last equality utilizes the fact that if the maximum on-chip nois ΔV_{max} is below threshold t, then the noise at any node must be below it.

Since for a given design, $P(\Delta V_{max} \leq t)$ is a constant independent of the sensor placement, it becomes clear that, in order to minimize the miss rate, it is equivalent to minimizing the following metric

$$P(\max(\Delta V_{r_i}(1 \leq i \leq s)) \leq t) \tag{1.4}$$

Or alternatively, it is equivalent to maximize

$$P(\max(\Delta V_{r_i}(1 \leq i \leq s)) \geq t) \tag{1.5}$$

Accordingly, the following definition can be derived for the Sensing Quality Metric (SQM).

Definition 1: The Sensing Quality Metric (SQM) for a set of nodes to be sensed is defined as the probability of the maximum noise among them goes beyond the threshold voltage as in Equation (1.5).

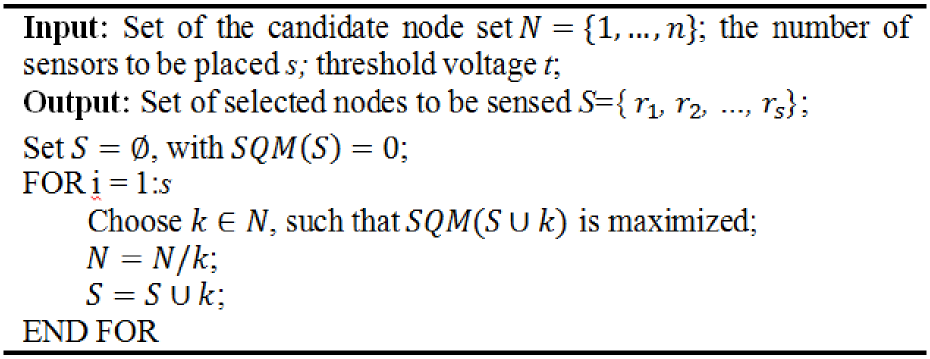
Intuitively, the SQM can be interpreted as the probability that a voltage emergency can be detected for the given sensor placement. In order to compute SQM, it is needed to evaluate the statistical max of a set of correlated random variables. Such techniques have been developed in the recent statistical timing analysis and power analysis research, for example, [37] for Gaussian random variables and [7] for non-Gaussian ones. The essence of those techniques is to represent the random variables as a function of the underlying common sources of process variations such as in (1.1).

1.4. ALGORITHM

1.4.1. Overview. Considering SQM, the objective turns to find s nodes among n candidates so that SQM of the set S of all selected nodes is maximized. A nave approach would be to enumerate all the possible combinations, which results in a complexity of $\binom{n}{s}$, which is exponential to s. Apparently, this is not a feasible method for large n and relatively large s in the problem.

To balance the complexity and quality of the solution, it is resorted to a compact greedy method, as shown in Figure 1.3. Basically, during each iteration, it always selects a node that could maximally increase the SQM of the set of selected nodes so far. By propagating the canonical form as shown in Equation (1.1), SQM($S \cup k$) could be easily calculated with the statistical max operation [7]. Since there are totally s iterations and the maximum number of SQM to be calculated is n, the algorithm complexity could be easily analysed as $O(sn)$. The simplicity of this algorithm prompts us to ask the following question: how good is the resulted solution? Will it be far from the optimal solution? In Section 1.4.2, the proof of optimality will be presented.

1.4.2. Proof of Optimality. In this section, the greedy method shown in Figure 1.3 is fomally proved to be the best-possible polynomial complexity approximation algorithm for maximizing the SQM.



```
Input: Set of the candidate node set $N = \{1, \ldots, n\}$; the number of sensors to be placed $s$; threshold voltage $t$;
Output: Set of selected nodes to be sensed $S$=\{ $r_1$, $r_2$, ..., $r_s$\};
Set $S = \emptyset$, with $SQM(S) = 0$;
FOR i = 1:$s$
    Choose $k \in N$, such that $SQM(S \cup k)$ is maximized;
    $N = N/k$;
    $S = S \cup k$;
END FOR
```

Figure 1.3: Proposed algorithm for noise sensor placement

The proof is inspired by a different interpretation of the SQM. Represented by the function form such as in Equation (1.1), the statistical noise of each node can be seen to lie in the variation space $\Omega(\delta Y)$, which is spanned by the random variables corresponding to global and local variation as defined in (1.1).

As such, the SQM of node i actually defines the mapping from Δv_i to the subspace $\omega_i \subseteq \Omega$ as

$$\omega_i = \{\Delta Y | \Delta v_i = H(\Delta G) + \Delta R_i \leq t\} \tag{1.6}$$

In other works, the SQM of a single node covers a subspace in Ω, and the SQM of a set of nodes $T = (n_i(1 \leq i \leq s)$ is the union of subspaces occupied by each of there nodes.

$$\omega_{SQM} = \cup \omega_{n_i}(1 \leq i \leq s) \tag{1.7}$$

Thus, SQM can also be quantitatively evaluated as the portion of the entire variation space occupied by selected nodes

$$SQM(T) = \frac{|\omega_{SQM}|}{|\Omega|} \tag{1.8}$$

where $|\bullet|$ is the Lebesgure measure (i.e., probability-weighted volume of the variation space).

From (1.7) and (1.8), it is clear that maximizing SQM(T) is equivalent to solving the *max variation space cover* (MVSC) problem, where it is needed to find a set of subspaces whose union covers the maximum volume in the variation space. The algorithm in Figure 1.3 can then be interpreted as to select a subspace for maximum incremental coverage at each iteration.

A related problem, the max set cover (MSC) problem [10], has been extensively studied and proven to be NP-hard. The MSC problem can be stated as follows: Given a set $M = a_i, , a_n$, a collection H composed of subsets $l_i \subseteq M$ and an integer k, the MSC problem tries to select k elements from H such that they cover the maximum number of elements in M. However, the proofs and conclusions in the MSC problem do not directly apply to the MVSC problem, as the former is defined in discrete space, while the latter works in continuous space. Similar to MSC problem, the MVSC problem is also NP-Hard. It will be showed in the following that that the proposed algorithm in Figure 1.3 can achieve the best polynomial approximation bound.

Corollary 1: The best polynomial-complexity approximation of MVSC is at most $1 - \frac{1}{e}$.

Proof: It comes from that the MVSC problem is the super-set of the MSC problem, and the MSC problem has best polynomial-complexity approximation $1 - \frac{1}{e}$ [10].

Next, the optimality of the greedy algorithm in Figure 1.3 will be proved using the variation space interpretation. As shown in Figure 1.4, OPT can be defined as the maximally covered variation space by the optimally placed s noise sensors through a black-box algorithm. ω_i is the space covered by the sensor placed in the i-th iteration of the algorithm in Figure 1.3, and ξ_i is the space that has not been covered after the i-th iteration, i.e.,

$$|\xi_i| = |OPT| - \sum_{j=1}^{i} |\omega_j| \tag{1.9}$$

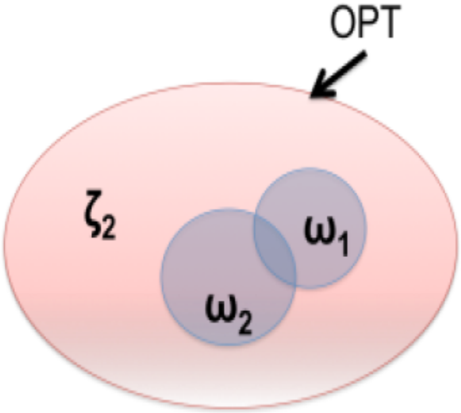


Figure 1.4: Variation space coverage of SQM

The algorithm in Figure 1.3 has the following properties.

Lemma 1: $|\omega_{i+1}| \geq \frac{|\xi_i|}{s}$.

Proof: Since the optimal solution uses s subspaces to cover OPT, after i-th iteration, there must exist an unselected subspace $\omega_{i+1}{'}$ covering at least $\frac{1}{s}$ fraction of ξ_i (otherwise ξ_i and thus OPT cannot be covered by the union of any s subspaces). Since in the $(i+1)$-th iteration the algorithm selects the subspace which maximally reduces the uncovered space, the selected subspace ω_{i+1} should reduce as much uncovered subspace as $\omega_{i+1}{'}$ does. Thus, it can be concluded that ω_{i+1} should have a volume of at least $\frac{|\xi_i|}{s}$.

Lemma 2 : $|\xi_{i+1}| \leq (1 - \frac{1}{s})^{i+1}|OPT|$

Proof: By induction. For i = 0, Lemma 2 is true; Assume Lemma 2 is true for i, i.e.,

$$|\xi_i| \leq (1 - \frac{1}{s})^i |OPT| \tag{1.10}$$

Thus from Lemma 1,

$$|\xi_{i+1}| = |OPT| - \sum_{j=1}^{i+1} |\omega_j| = |\xi_i| - |\omega_{i+1}| \leq |\xi_i| - \frac{|\xi_i|}{s} \leq (1 - \frac{1}{s})^{i+1}|OPT| \tag{1.11}$$

From Lemma 1 and Lemma 2, the following theorem can be proved.

Theorem 2: The algorithm in Figure 1.3 is $(1-\frac{1}{e})$ approximation of OPT.

Proof: Replace s with $i+1$ in Lemma 2,

$$|\xi_s| \leq (1-\frac{1}{s})^s|OPT| \leq \frac{|OPT|}{s} \tag{1.12}$$

From Corollary 1 and Theorem 2, it is straightforward to get the following corollary.

Corollary 2: The proposed greedy algorithm in Figure 1.3 is the best-possible polynomial time algorithm for maximizing SQM.

1.5. EXPERIMENTAL RESULTS

The proposed greedy algorithm in Figure 1.3 is implemented in C++ on a machine with two quad-core 2.4 GHz Intel Xeon E5620 CPUs and 96 GB memories. A similar method as [37] is adopted to obtain the statistical noise model Z_i for Vdd nets N1, N2 and N3, which are extracted from real industrial power grid designs. As summarized in Table 1.1, the #i, #n and #r stand for the number of current sources, the number of nodes and the number of resistors of each net, respectively. In addition, it is assumed that all the noises are Gaussian, and apply PCA to get the linear canonical form [37]. A zero-mean Gaussian random variable ΔR_i is further added to model the independent noise variation of each node which may come from PCA model error, noise sensor process variation, fluctuations in reference voltage, etc. In the experiments, threshold time T_t is set to 1ns. To compare the quality of the sensor placement result, three alternative noise sensor placement techniques are implemented. The first method simply selects the top-s nodes with maximum

Table 1.1: Benchmark information

Benchmark	#i	#n	#r
N1	5,387	5,387	4,720
N2	18,419	19,240	38,366
N3	100,527	102,178	197,470

average noise for sensor placement (denoted as Ave-noise), while the second and third methods map noise to temperature and employ the state-of-the-art temperature sensor allocation techniques Eigenmap [31] and [32], respectively. Note that the former targets at recovering the whole noise map, while the later targets at capturing the hot spots at all times.

Table 1.2 compares the miss rate of the three methods. The miss rate is directly calculated as p/q, where q is the total number of time steps in the transient noise waveforms and p is the number of time steps in which a voltage emergency is missed (i.e., the placed sensors fail to detect the voltage emergency). The 90 mV threshold is set as 5% of the nominal Vdd (1.8V) according to the reported hard threshold value in literature [11]. The results in Table 1.2 indicate that compared with Ave-noise, the method in [32] and EigenMap [31], Eagle-Eye on average reduces the miss rate of voltage emergency detections by 7.4x, 6.2x and 15x, respectively. The drastic improvement should mainly be credited to the difference in optimization objectives between these methods. The runtime comparison in Table 1.3 also shows that Eagle-Eye is on average 96x faster than EigenMap and 3.2x faster than [32]. Such runtime reduction comes from the fact that the statistical max operations used in Eagle-Eye is much faster than the complex matrix operations needed in EigenMap and [32]. Figure 1.5 shows how the SQM changes with the number of placed sensors under different threshold voltages. In addition to the hard threshold (5%Vdd, 90mV), two soft thresholds 87 mV and 85 mV are also used. Based on the observation in (1.5), it is natural to see that, under all thresholds, SQM increases with the number

Table 1.2: Miss rate comparisons (#sensor = 10, t = 90mV)

Benchmark	Miss Rate			
	Ave-noise	Reda, S	EigenMap	Eagle-Eye
N1	17%(8.5x)	19%(9.5x)	75%(38x)	2%(1x)
N2	38%(19x)	30%(15x)	47%(24x)	2%(1x)
N3	18%(3.0x)	13%(2.2x)	29%(4.8x)	6%(1x)
Average	24.3%(7.4x)	20.6%(6.2x)	50.3%(15x)	3.3%(1x)

Table 1.3: Runtime comparisons (#sensor = 10, t = 90mV)

Benchmark	Runtime(sec)			
	Ave-noise	Reda, S	EigenMap	Eagle-Eye
N1	0.01 (1/14x)	0.25 (1.8x)	1.68 (12x)	0.14 (1x)
N2	0.01 (1/20x)	0.67 (3.4x)	27.12(136x)	0.20(1x)
N3	0.03 (1/10x)	1.03 (3.4x)	31.65 (106x)	0.30 (1x)
Average	0.02 (1/11x)	0.65 (3.2x)	20.15 (96x)	0.21 (1x)

of sensors. In all test cases, with fewer than 10 sensors, SQM reaches 90% of its maximum. This suggests that only a small number of sensors are indeed necessary, which means the framework will induce very little hardware overhead. The miss rate also drops with the number of sensors as expected. Interestingly, it is observed a rapid drop of miss rate at certain knee points of the number of noise sensors. This suggests the existence of a few nodes in the given power grid design, the union of which can be used as a good statistical indicator for the full-chip voltage emergencies. The number of nodes needed to form such a representative noise indicator increases as the threshold decreases. It is also found from this experiment that there is no overlap between those groups of nodes being selected at different thresholds.

Figure 1.6 compares the miss rate of different approaches with different number of noise sensors at 90 mV threshold on N2. Apparently, with the increase of the sensor number, the miss rate of Eagle-Eye quickly drops below 2%, while those of the Ave-noise and [32] are much slower. This is because [32] focuses on capturing the maximum noise locations at all times. When the number of sensors is limited, it may not select the nodes that have less noise but are better indicators of full-chip voltage emergencies. Even worse, the miss rate of EigenMap fluctuates extensively. This is because EigenMap emphasizes more on recovering the entire noise map. As such, it fails to capture the representative nodes as they are relatively uncorrelated to the other power grid nodes in the design.

It is further studied how SQM and miss rate change as the threshold voltage increases, and the result is shown in Figure 1.7 for N2 with 10 sensors. It is interesting

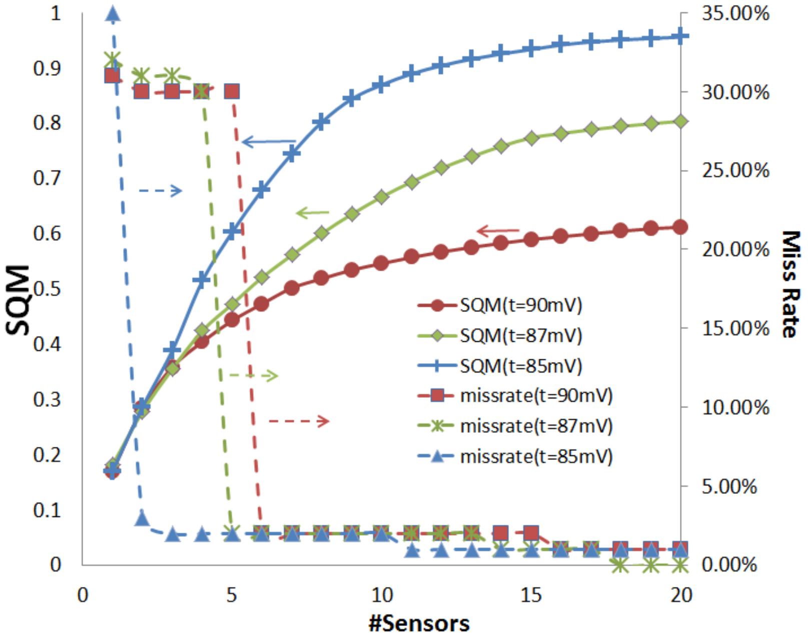


Figure 1.5: SQM and miss rate vs. the number of sensors of N2

that although SQM decreases in general with the increase of threshold voltage, which is intuitive from (1.5), the miss rate fluctuates with the overall trend of decreasing. The sudden drop of miss rate can be noted, again due to the existence of a few nodes whose union form a good statistical indicator for full-chip voltage emergencies for 85 mV threshold voltage and above.

Finally, for the same benchmark N2, sensor number 10 and threshold voltage 90 mV, Figure 1.8 studies the impact of sensing inaccuracy (i.e., ΔR_i in (1.1) induced by factors such as measurement error and process variations of the sensors) on the final sensor placement result. Here ΔR_i is represented as a zero-mean Gaussian random variable. Obviously, when the sensing inaccuracy is small ($\leq 20\%$), the miss rate and SQM remain flat, which shows the Eagle-Eye has strong resistance against small sensing inaccuracy. However, as the sensing inaccuracy increases, the chip noises become largely randomized. As such, it is harder to select a good statistical indicator for full-chip voltage emergencies, which translates into a rapid increase in the miss rate. The increase in SQM comes from the increased noise variations. Although not

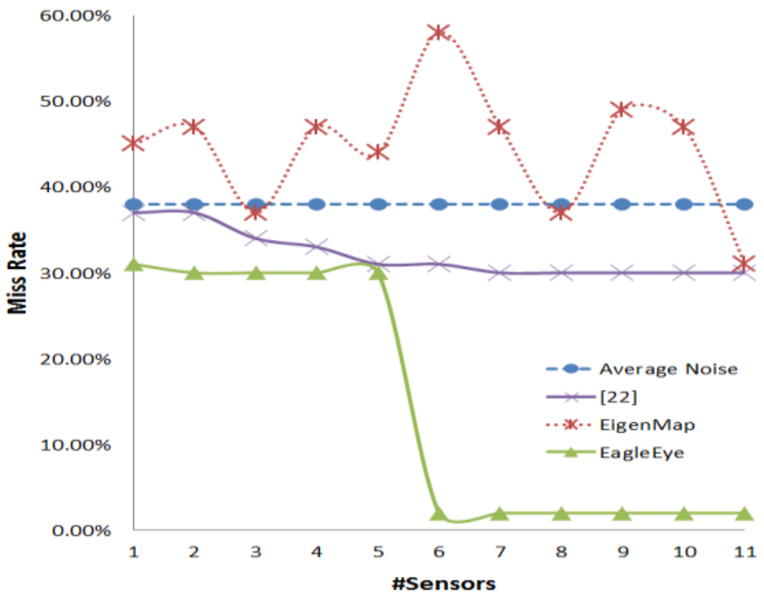


Figure 1.6: Miss rate vs. the number of sensors for different approaches (threshold = 90 mV, N2)

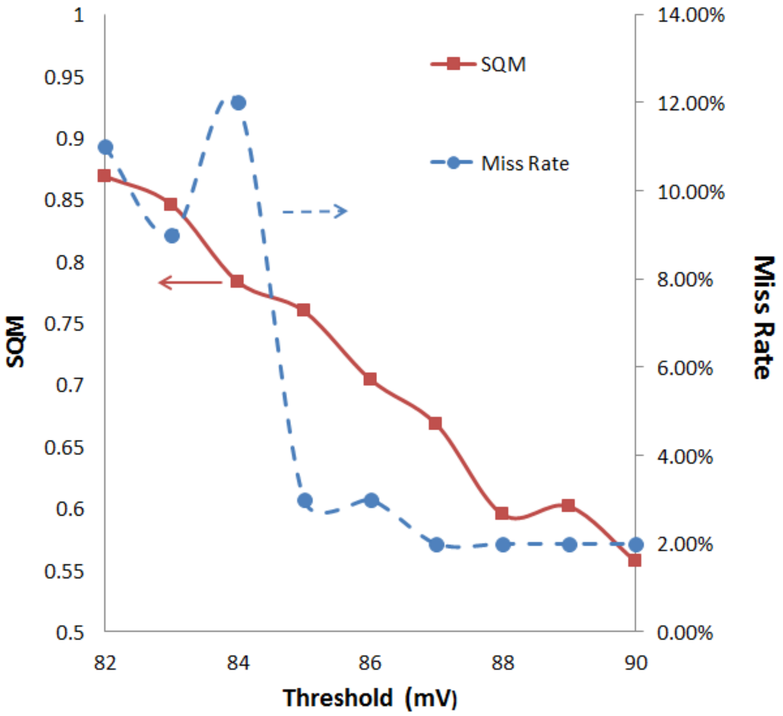


Figure 1.7: SQM and miss rate vs. threshold (number of sensors = 10, N2)

shown, it is also observed that the resistance to sensing inaccuracy becomes stronger as the sensor number increases.

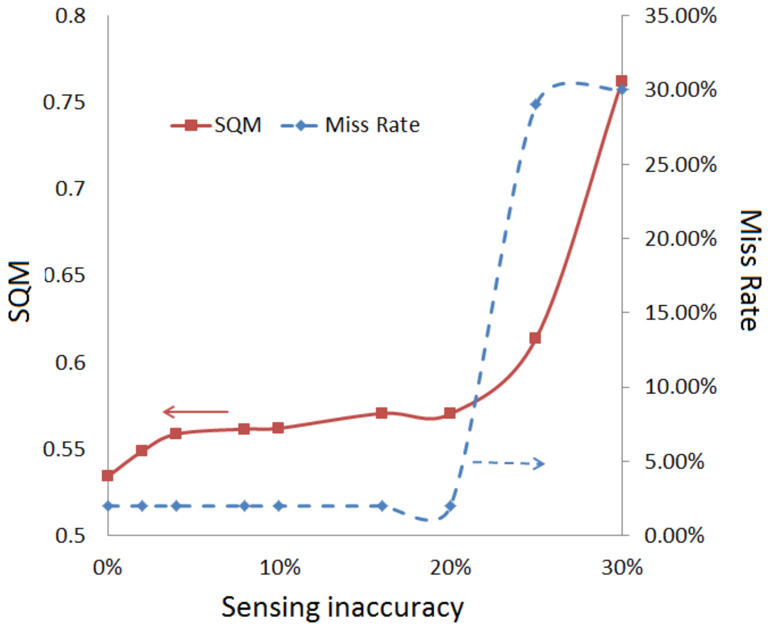


Figure 1.8: SQM and miss rate vs. sensing inaccuracy (threshold = 90 mV, # of sensors = 10, N2)

1.6. CONCLUSION

In this chapter, a compact, fast and near-optimal solution for noise sensor placement is proposed. It can help to detect voltage emergencies efficiently and provide sensor placement strategy for runtime power management systems. Experimental results on a set of industrial power grid designs show that compared with a simple average-noise based heuristic and two state-of-the-art temperature sensor placement algorithms aiming at recovering the full map or capturing the hot spots at all times, the proposed method on average can reduce the miss rate of voltage emergency detections by 7.4x, 15x and 6.2x, respectively.

2. ON THE OPTIMAL THRESHOLD VOLTAGE COMPUTATION OF ON-CHIP NOISE SENSORS

2.1. ABSTRACT

Runtime noise management systems typically rely on on-chip noise sensors to accurately capture voltage emergencies. As such, the threshold voltage for noise sensors to report emergencies serves as a critical tuning knob between the system failure rate and false alarms. Unfortunately, the problem of optimal threshold voltage computation remains open in literature despite its importance. The problem is further complicated by process variations, which introduce significant variations in load currents and thus in noise across different chips. A uniform noise margin may not work optimally for all the chips. In this chapter, the problem of minimizing the system alarm rate subject to a given system failure rate constraint is first formulated. A uniform scheme is then put forward to find an optimal solution for all chips. Compared to a seemingly more intuitive approach which is too conservative, experimental results over a set of industrial designs show an average of 20.6% reduction in system alarm rate under the same system failure rate constraint. It is further shown that with the help of I_{ddq} measurements during testing which reveal process variation information, it is possible and efficient to compute a per-chip optimal threshold voltage threshold. It further reduce the alarm rate by 12.3% on average compared with uniform threshold approach.

2.2. INTRODUCTION

The relentless CMOS scaling has increased the power density drastically, suppressing the chip supply voltage. On the other hand, to avoid excessive leakage power, the threshold voltage of transistors cannot be scaled at the same pace. As a result, it is becoming increasingly difficult to ensure power integrity through design time techniques such as decoupling capacitance budgeting [35], power grid sizing [8], etc.

To alleviate the problem, microarchitecture level noise management systems have been studied recently, trying to address power integrity issues at runtime. In most of these systems, noise sensors are deployed to detect voltage emergencies, which are defined as the situations when the amplitude of the noise exceeds a given *threshold voltage* V_t for a minimum duration (*threshold time*) of T_t [16]. Once a voltage emergency is detected, system level controls such as instruction throttling [16, 27] or commit-and-rollback [33] can be applied to ensure the correct operation of the system.

Two things play central roles in these runtime noise management systems: the placement of the noise sensors, as well as the threshold voltage V_t and threshold time T_t to decide the occurrence of voltage emergencies. Near-optimal noise sensor placement method is proposed in [41], which provides a promising solution to the first problem. However, the second problem still remains largely unexplored in the literature. As such, these thresholds are normally decided empirically by experienced designers.

As threshold voltage and threshold time have a combined effect on circuit performance, in this chapter it assumes that threshold time is given and focus on the optimal decision of threshold voltage. There are two types of threshold voltages. The *hard threshold* t_h is the strict bound which is fixed once a chip is fabricated. When the noise exceeds this bound for a duration of at least T_t, the system malfunctions. In this chapter, this type of voltage emergencies is called as *real voltage emergencies.* On the other hand, since it can only place a limited number of noise sensors on chip and the sensors are not always accurate, not all real voltage emergencies can be detected. Also the system takes time to respond to the voltage emergencies reported by sensors. Accordingly, a *soft threshold* t_s, which is lower than t_h so as to leave enough voltage emergency detection margin, is typically used by the sensors to serve as an alarm for potential real voltage emergency and system failure. In other words, the system uses t_s to report voltage emergencies while the real voltage emergency (system malfunction) actually happens when t_h is crossed.

As will be shown in Section 2.3.2, the value of t_s has large impacts on the system performance. If t_s is set too high (i.e., close to t_h), the system may suffer from large system failure rate, i.e., the undetected voltage emergencies can potentially cause system failures, resulting in reboots. Here the *system failure rate* is defined as the probability that the placed noise sensors do not detect any voltage emergencies while the maximum noise on chip is over t_h, i.e., a real voltage emergency occurs. On the other hand, if t_s is set too low, the chance of false alarms (i.e., the noise crosses t_s but not t_h) increases. Since the system relies on the sensor signals to act, these false alarms will introduce runtime performance loss (RPL) to handle the nonexistent real voltage emergencies. It is thus imperative to identify an optimal t_s that balances the system failure rate and RPL.

The problem is further complicated by the process variations across chips as well as environmental uncertainties. Variations introduce significant difference in load currents and thus in noise across different chips. On the other hand, they also introduce difference in critical path delay, and thus in t_h. In other words, a uniform threshold voltage may not work well for all chips and it is better to be calculated on a per-chip basis. Moreover, noise sensors themselves also suffer from on-chip noise coupling and other environmental uncertainties, which may give inaccurate sensing results. As such, it is needed to take these uncertainties into consideration when computing the optimal threshold voltage.

In this chapter a statistical approach is proposed to compute the optimal soft threshold voltage t_s, under which the alarm rate is minimized while satisfying the user supplied system failure rate constraint. In detail, two scenarios are analyzed. In the first scenario, a single uniform t_s is caculated for all chips with the same design. Compared to a seemingly more intuitive approach which is too conservative, experimental results show 20.6% average alarm rate reduction over a set of industrial designs under the same system failure rate constraint. In addition, considering the existence of process variations, a per-chip optimal t_s is further proposed based on the chip leakage current I_{ddq} measurement during testing, assuming fixed t_h. Experimental results show that such an approach further reduces the alarm rate by 12.3% compared with the uniform t_s approach under the same system failure rate constraint. To the best of our knowledge, this is the first work that systematically formulates and solves the optimal soft noise threshold voltage computation problem for on-chip noise sensors.

The rest of this chapter is organized as follows. Backgrounds and preliminaries are introduced in Section 2.3. The problem of optimal threshold voltage computation formally formulate in Section 2.4. Section 2.5 provides a method to compute a uniform soft threshold voltage for all chips, while the per-chip threshold voltage computation is illustrated in Section 2.6. Section 2.7 shows the experimental results by comparing the different threshold computation methods. Conclusions are given in Section 2.8.

2.3. PRELIMINARIES

2.3.1. Runtime Noise Management Systems. Runtime noise management systems at microarchitecture level are normally composed of a centralized controller and distributed noise sensors. Voltage emergencies are detected by comparing the sensed noise with a threshold voltage over a period of threshold time. In general, these systems can be classified into three categories: retroactive systems with hard threshold voltage t_h [33], retroactive systems with soft threshold voltage t_s [16, 27] and proactive systems with soft threshold voltage t_s [11].

The retroactive systems with hard threshold voltage works by setting the noise sensor threshold to strict t_h [33]. Once t_h crossing is detected (i.e., real voltage emergency occurs), the system needs to be recovered to its previous correct status through mechanisms such as commit-and-rollback. Otherwise the system will malfunction due to the excessive noise. To avoid the large overhead introduced by full system rollback, recently the retroactive system with soft threshold voltage [16, 27] has been proposed. By detecting the less strict t_s crossing, the system signals alarms for potential t_h crossing. When such an alarm is reported, the system can take immediate actions such as instruction throttling to prevent real voltage emergencies from happening and thus avoiding the overhead from system rollback. By leaving proper margins between t_s and t_h, such systems can be kept running without real voltage emergencies. The proactive system with soft threshold [11] takes one step further, which tries to recognize and track the patterns of activities that may lead real voltage emergencies, and invoke the throttling mechanism based on the prediction in advance to prevent voltage emergencies.

This chapter focuses on systems using soft threshold voltage t_s since they incur less system overhead, and propose statistical techniques to decide the optimal value for t_s.

2.3.2. Impact of Soft Threshold Voltage. This section provides readers with a better understanding of the impact of t_s on voltage emergency detection quality and system performance, through a graphical approach. Figure 2.1 shows the joint probability density function (JPDF) of the chip max noise Z_{max} (i.e., fixed once the chip is fabricated) and sensed max noise Z_S (i.e., fixed once the sensor placement is given). The ellipses in Figure 2.1 represent contours of equal probability. The joint distribution can be obtained by regression on samples from simulations.

The horizontal line represents the hard threshold voltage t_h, i.e., $Z_{max} = t_h$. The dashed vertical line represents the soft noise threshold voltage, which crosses the horizontal axis at t_s, i.e., $Z_S = t_s$. As such, the area below (above) the horizontal line has $Z_{max} < t_h$ ($Z_{max} > t_h$), i.e., no real voltage emergency occurs (real voltage emergency occurs). In addition, based on t_s, the area to the left of the dashed vertical line is where the sensors will not report an alarm, while the area to the right of this line is where alarms will be signaled by the sensors. As such, four regions are evident:

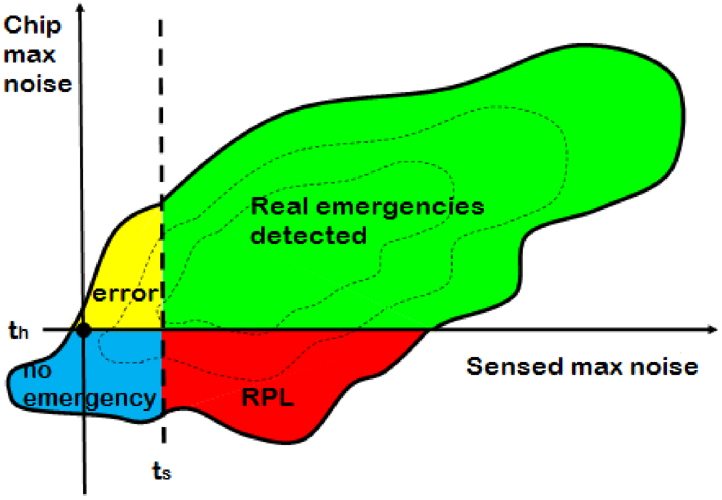


Figure 2.1: Joint distribution of the chip maximum noise and sensed maximum noise

1) The region marked with *no emergency* represents the safe cases, i.e., no voltage emergencies detected by the sensors, and no real voltage emergencies occur anywhere on the chip.

2) The region marked with *real emergencies detected* comprises the cases where real voltage emergencies actually happen and the sensors signal alarms for voltage emergencies.

3) The region marked with *error* represents the situations where the sensors fail to detect real voltage emergencies. Such miss will lead to system failures and should be minimized.

4) The region marked with *RPL* contains the cases when sensors signal alarms for voltage emergencies while no real voltage emergency is occurring anywhere on the chip. These are false alarms that will introduce system overhead by triggering unnecessary noise management procedures such as instruction throttling. As a result, it is desirable to limit such performance loss due to false alarms under a certain level.

From Figure 2.1 it can be seen that as the soft threshold increases (the dashed vertical line moves towards right), the chances of system failure (the *error* region) increases but the RPL due to false alarm decreases (and vice versa). To the extreme, when RPL = 0, $t_s = t_h$. While the tradeoff can be considered in many different ways, this chapter considers the problem of minimizing the chances of alarm rate to avoid runtime performance loss, while the system failure rate is constrained by a user-specified level.

2.4. PROBLEM FORMULATION

It is difficult to analyze the voltage emergency when both threshold voltage and threshold time are coupled together. As such, a method to transform the noise waveform $\Delta V_i(t)$ of node i at time t is utilized, which can be obtained from SPICE simulation, as follows:

$$z_i(t) = \min_{t-T_t \leq t' \leq t} \Delta V_i(t') \tag{2.1}$$

In other words, at each time instance, the minimum of the noise is taken within the time window of length T_t backward. The reason to use such transform is that the criterion to check for voltage emergencies now becomes very simple: it occurs at node i if and only if at some time instance t_0, $z_i(t_0)$ is higher than the threshold voltage V_t*. In other words, the transformation in (2.1) allows to include threshold time T_t implicitly.

One challenging problem is to quantitatively evaluate the *system failure rate* with given noise sensor placements. As it is impossible to obtain all possible voltage droop (noise) waveforms and thus all possible z_i waveforms at a node i through transient simulation, same as [38] it is proposed to model the variation of z_i over time, as a random variable Z_i:

$$Z_i = F_i(X) = H_i(G) + \Delta R_i \tag{2.2}$$

*Note V_t can be either soft or hard threshold voltage. In this chapter, $V_t = t_s$ is used for voltage emergency detection, and $V_t = t_h$ is used to decide if an real voltage emergency has occurred.

where X is a set of common correlated factors that lead to the noise variation through function F_i. Through modeling techniques, the noise can be represented by function $H_i(G)$, where G is an m-dimensional uncorrelated random variable that models the global variation sources (common for all nodes) which can be extracted from X through principle component analysis (PCA). The dimension m decides the approximation accuracy. ΔR_i models the independent source of noise variation specific to node i which comes from model error, noise sensor process variation, fluctuations in reference voltage, etc. Linear form of F_i and H_i are used in this chapter, however, this is not a limitation of the proposed method.

Note that in this chapter it is assumed that the noise follows Gaussian distributions. It has been shown in [9] that this in general holds except near the tail when the noise is small, which will introduce some minor errors in our scheme. The impact of this error is revealed in some of the experiments to be reported later.

From the analysis in Section 2.3.1, as long as the sensors signal an alarm, the system will react and cause performance degradation. Accordingly, it is of interest to minimize the chances that the sensors signal alarms, or the *alarm rate*. Considering the tradeoffs revealed in Section 2.3.2, it is proposed to minimize the alarm rate under a given system failure rate. With the above noise model, to best utilize the runtime noise management system, it is needed to minimize the alarm rate under a given system failure rate constraint. With the above noise model, the problem of optimal threshold voltage computation can be formulated as follows: given the chip max noise Z_{max} and the sensed max noise $Z_S = max(Z_{r_i}(1 \leq i \leq s))$ (i.e., $r_1, r_2, \ldots, r_s$ are the s nodes with pre-placed noise sensors) distributions, hard threshold t_h and a maximum allowed system failure rate q specified by users, compute the soft threshold voltage t_s as the solution of the optimization problem

$$\arg\min_{t_s} P(Z_S \geq t_s) \tag{2.3}$$

$$\text{s.t. } P(Z_{max} \geq t_h | Z_S \leq t_s) \leq q. \tag{2.4}$$

where $P(Z_S \geq t_s)$ represents the *system alarm rate* (i.e., the probability of the soft threshold crossing), and *system failure rate* is expressed as $P(Z_{max} \geq t_h | Z_S \leq t_s)$ (i.e., the probability of undetected real voltage emergencies when the soft threshold of sensors is not crossed).

2.5. UNIFORM SOFT THRESHOLD COMPUTATION

In this section, two algorithms are presented to solve the above problem efficiently.

2.5.1. An Intuitive Approach. The intuitive approach is to consider the statistical difference $\Delta Z = Z_{max} - Z_S$ between the chip max noise and the sensed max noise. When the sensors miss a voltage emergency, $Z_S \leq t_s$, i.e., t_s is the higher bound for Z_S. Consequently,

$$\begin{aligned} &P(Z_{max} \geq t_h | Z_S \leq t_s) = \\ &P(Z_S + \Delta Z \geq t_h | Z_S \leq t_s) \leq P(t_s + \Delta Z \geq t_h | Z_S \leq t_s) \end{aligned} \tag{2.5}$$

However, since ΔZ is correlated to t_s, this gives no straightforward way to calculate the feasible t_s.

To solve this problem, it is noticed that the chip max noise can be decomposed into a linear combination of a part that is correlated to sensed max noise, and a part that is uncorrelated as follows

$$Z_{max} = \alpha Z_S + \Delta Z_S \tag{2.6}$$

where ΔZ_s is uncorrelated with the sensed max noise. α is proportional to the correlation between Z_{max} and Z_S, and lies between 0.5 and 0.8 in our experiments.

As a result, the system failure rate can be derived as

$$\begin{aligned} P(Z_{max} \geq t_h | Z_S \leq t_s) &= \frac{P(Z_{max} \geq t_h, Z_S \leq t_s)}{P(Z_S \leq t_s)} \\ =&\frac{P(\alpha Z_S + \Delta Z_S \geq t_h, Z_S \leq t_s)}{P(Z_S \leq t_s)} \leq \frac{P(\alpha t_s + \Delta Z_S \geq t_h, Z_S \leq t_s)}{P(Z_S \leq t_s)} \\ =&\frac{P(\alpha t_s + \Delta Z_S \geq t_h) P(Z_S \leq t_s)}{P(Z_S \leq t_s)} = P(\alpha t_s + \Delta Z_S \geq t_h) \end{aligned} \tag{2.7}$$

In order to satisfy *system failure rate* $\leq q$, it is equal to have $P(\alpha t_s + \Delta Z_S \geq t_h) \leq q$, and therefore a conservative estimate for the soft threshold is

$$t_s = \frac{1}{\alpha}(t_h - \sigma_s \Phi^{-1}(q) - \mu_s) \tag{2.8}$$

where Φ^{-1} represents the inverse of the CDF of a unit Gaussian, and μ_s , σ_s are the mean and standard deviation of ΔZ_S. However, although (2.8) gives a feasible solution to the problem, the solution is too conservative and thus sub-optimal. In the next section, an exact method is proposed to generate a statistical optimal solution for the uniform t_s calculation.

2.5.2. An Exact Approach. As the chip max noise and the sensed max noise can both be expressed in the form of (2.2), the uniform soft threshold can be computed exactly.

Denote $Q = P(Z_{max} \geq t_h | Z_S \leq t_s)$, and have

$$\begin{aligned} Q &= P(Z_{max} \geq t_h | Z_S \leq t_s) \\ &= \frac{P(Z_{max} \geq t_h, Z_S \leq t_s)}{P(Z_S \leq t_s)} \\ &= \frac{\int_{-\infty}^{t_s} \int_{t_h}^{\infty} p_c(Z_{max}, Z_S) dZ_{max} dZ_S}{\int_{-\infty}^{t_s} p_t(Z_S) dZ_S} \end{aligned} \tag{2.9}$$

The derivative of the constraint function (2.4) with respect to t_s is

$$\frac{dQ}{dt_s} = \frac{A'B - AB'}{B^2} = \frac{A'}{B^2}(B - \frac{AB'}{A'}) \tag{2.10}$$

where

$$\begin{aligned} A &= \int_{-\infty}^{t_s} \int_{t_h}^{\infty} p_c(Z_{max}, Z_S) dZ_{max} dZ_S \\ &= \int_{-\infty}^{t_s} \left(p_t(Z_S) \int_{t_h}^{\infty} p_c(Z_{max}|Z_S) dZ_{max} \right) dZ_S \end{aligned} \tag{2.11}$$

$$B = \int_{-\infty}^{t_s} p_t(Z_S) dZ_S \tag{2.12}$$

$$A' = \frac{dC}{dt_s} = p_t(t_s) \int_{t_h}^{\infty} p_c(Z_{max}|t_s) dZ_{max} \tag{2.13}$$

$$B' = \frac{dB}{dt_s} = p_t(t_s) \tag{2.14}$$

Using (2.11)(2.13)(2.14), $\frac{AB'}{A'}$ can be expressed as

$$\frac{AB'}{A'} = \int_{-\infty}^{t_s} p_t(Z_S) \frac{\int_{t_h}^{\infty} p_c(Z_{max}|Z_S) dZ_{max}}{\int_{t_h}^{\infty} p_c(Z_{max}|t_s) dZ_{max}} dZ_S \tag{2.15}$$

In the above equations, $p_c(Z_{max}|Z_S)$ means the conditional probability density of Z_{max} at the given value of Z_S. $p_c(Z_{max}|t_s)$ implies $p_c(Z_{max}|Z_S = t_s)$. In this way, $\frac{AB'}{A'}$ can be expressed as

$$\begin{aligned} &\int_{t_h}^{\infty} p_c(Z_{max}|Z_S) dZ_{max} \\ &= 1 - \Phi \left(\frac{t_h - \mu_{max} - \rho \frac{\sigma_{max}}{\sigma_S} (Z_S - \mu_s)}{\sigma_{max} \sqrt{1 - \rho^2}} \right) \end{aligned} \tag{2.16}$$

$$\int_{t_h}^{\infty} p_c(Z_{max}|t_s) dZ_{max} = 1 - \Phi \left(\frac{t_h - \mu_{max} - \rho \frac{\sigma_{max}}{\sigma_S} (t_s - \mu_s)}{\sigma_{max} \sqrt{1 - \rho^2}} \right) \tag{2.17}$$

The two integrals (2.16) and (2.17) differ from each other only in the appearance of Z_S and t_s. As $Z_S \leq t_s$, it can be concluded that:

$$0 < \frac{\int_{t_h}^{\infty} p_c(Z_{max}|Z_S)dZ_{max}}{\int_{t_h}^{\infty} p_c(Z_{max}|t_s)dZ_{max}} \left\{ \begin{matrix} < 1, if \rho > 0 \\ = 1, if \rho = 0 \\ > 1, if \rho < 0 \end{matrix} \right\} \tag{2.18}$$

From (2.18), it is clear that $(B - \frac{AB'}{A'})$in (2.10) is positive (negative) when ρ is negative (positive). From (2.13), $A^{'} > 0$ as well. Combining these two conclusions, it can be seen that the constraint function derivative in (2.10) is positive (negative) when ρ is positive (negative). The objective function is therefore also a monotone function of t_s.

Denote $R = P(Z_S \geq t_s)$, and have

$$R = \int_{t_s}^{\infty} p_t(Z_S)dZ_S \tag{2.19}$$

From (2.19), it is clear that the objective function is also a monotone function of t_s.

Both the objective function (2.3) and the constraint (2.4) are monotone functions of t_s. As such, the optimal soft threshold can be computed from the constraint

$$P(Z_{max} \geq t_h|Z_S \leq t_s) = q \tag{2.20}$$

Rewriting (2.9), it can be obtained that

$$\int_{-\infty}^{t_s} \int_{t_h}^{\infty} p_c(Z_{max}, Z_S)dZ_{max}dZ_S = q \int_{-\infty}^{t_s} p_t(Z_S)dZ_S \tag{2.21}$$

Taking into consideration that $p_c(Z_{max}, Z_S)$ and $p_t(Z_S)$ are Gaussian PDFs, this equation can be simplified. The double integral of the left-hand side can be reduced to a one-dimensional integral by analytic integration over Z_S and the right-hand side can also be integrated analytically. The resulting equation has only a single-dimensional integral, thus (2.21) can be solved numerically.

2.6. PER-CHIP SOFT THRESHOLD COMPUTATION

The uniform solution in Section 2.5 may not be optimal for all chips due to the existence of process variations and their impacts on noise margin [4, 25]. On the other hand, the leakage current I_{ddq}, which is easy to measure during testing, has been observed to have strong correlation with the chip max noise in literature [22].

As such, by measuring I_{ddq} value during testing stage, it is able to quantitatively evaluate or sample the per-chip max noise Z_{max} information under process variation. This is done by first building the correlation between I_{ddq} and Z_{max} through linear regression of Monte Carlo simulation results. Figure 2.2 illustrates the corresponding results in 45 nm technology, which further confirms the strong correlation between I_{ddq} and Z_{max}. During testing, the Z_{max} can then be easily decided from the measured I_{ddq} using this pre-built correlation. With such extra per-chip Z_{max} information, the optimal t_s for each chip can then be calculated based on the following derivations. Given the chip max noise Z_{max}, the sensed max noise $Z_S = max(Z_{r_i}(1 \leq i \leq s))$, and the I_{ddq} distribution Z_D in the form of (2.2), hard threshold t_h and a maximum allowed system failure rate q, for each value of $I_{ddq}(Z_D)$ compute the soft threshold t_s as the solution of the optimization problem

$$\arg\min_{t_s(Z_D)} P(Z_S \geq t_s(Z_D)) \tag{2.22}$$

$$s.t.\ P(Z_{max} \geq t_h | Z_S \leq t_s(Z_D)) \leq q. \tag{2.23}$$

The difference between (2.3)(2.4) and (2.22)(2.23) is that in (2.22)(2.23) the soft threshold is dependent on I_{ddq} measurement.

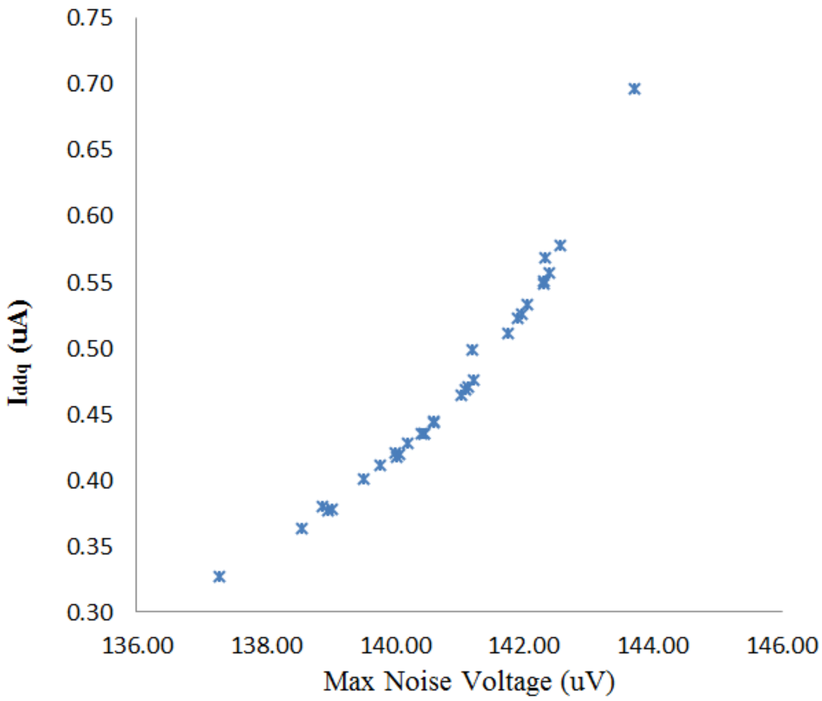


Figure 2.2: Relation between I_{ddq} and max noise. The experiment is done by HSPICE simulation of a NAND gate in 45nm node

2.6.1. Per-Chip Optimal Threshold Computation with Fixed Hard Threshold. Now both the objective function and constraint are functionals. This difference can be more obvious if reformulating the probabilities as integrals.

$$\arg\min \int_{-\infty}^{\infty} \int_{t_s(Z_D)}^{\infty} p_t(Z_S) dZ_S dZ_D \tag{2.24}$$

$$s.t. \ \frac{\int_{-\infty}^{\infty} \int_{-\infty}^{t_s(Z_D)} \int_{t_h}^{\infty} p_c(Z_{max}, Z_S, Z_D) dZ_{max} dZ_S dZ_D}{\int_{-\infty}^{\infty} \int_{-\infty}^{t_s(Z_D)} p_t(Z_S, Z_D) dZ_S dZ_D} \leq q. \tag{2.25}$$

The constraint (2.25) can be formulated in the form of an equality using the obvious fact that the optimal solution is achieved when system failure rate is exactly as required and no less. The constraint (2.25) can be rewritten as

$$\begin{aligned}&\int_{-\infty}^{\infty}\int_{-\infty}^{t_s(Z_D)}\int_{t_h}^{\infty} p_c(Z_{max}, Z_S, Z_D)dZ_{max}dZ_SdZ_D\\ &-q\int_{-\infty}^{\infty}\int_{-\infty}^{t_s(Z_D)} p_t(Z_S, Z_D)dZ_SdZ_D = 0.\end{aligned}\tag{2.26}$$

The objective (2.24) is equivalent to $\arg\max \int_{-\infty}^{\infty}\int_{-\infty}^{t_s(Z_D)} p_t(Z_S)dZ_SdZ_D$. Thus the Lagrangian can be written as

$$\begin{aligned}&\int_{-\infty}^{\infty}\int_{-\infty}^{t_s(Z_D)}((1+\lambda q)p_t(Z_S, Z_D)\\ &-\lambda\int_{t_h}^{\infty} p_c(Z_{max}, Z_S, Z_D)dZ_{max})dZ_SdZ_D\end{aligned}\tag{2.27}$$

where λ is the Lagrange multiplier.

$y(x)$ satisfies $\frac{\partial H}{\partial y} = 0$ when the functional $\int_{-\infty}^{\infty} H(x, y(x))dx$ reaches its optimum in variational calculus [30]. Therefore, for $t_s(Z_D, \lambda)$, when the optimum value is reached:

$$(1+\lambda q)p_t(t_s, Z_D) - \lambda\int_{t_h}^{\infty} p_c(Z_{max}, t_s, Z_D)dZ_{max} = 0. \tag{2.28}$$

(2.28) by $\lambda p_t(t_s, Z_D)$ and apply the formula for conditional probability, and getting

$$\int_{t_h}^{\infty} p_c(Z_{max}|t_s, Z_D)dZ_{max} = q + \frac{1}{\lambda}. \tag{2.29}$$

On the other hand, the vector of noise Z, vector of mean values μ and correlation matrix Σ of the JPDF $p_c(Z_{max}, Z_S, Z_D)$ are partitioned as follows

$$Z = \begin{pmatrix} Z_{max}\\ Z_S\\ Z_D\end{pmatrix} = \begin{pmatrix} Z_{max}\\ Z_{SD}\end{pmatrix} \tag{2.30}$$

$$\mu = \begin{pmatrix} \mu_{max} \\ \mu_S \\ \mu_D \end{pmatrix} = \begin{pmatrix} \mu_{max} \\ \mu_{SD} \end{pmatrix} \tag{2.31}$$

$$\begin{aligned} \Sigma &= \begin{pmatrix} \sigma^2_{max} & \rho_{max,S} & \rho_{max,D} \\ \rho_{max,S} & \sigma^2_S & \rho_{S,D} \\ \rho_{max,D} & \rho_{S,D} & \sigma^2_D \end{pmatrix} \\ &= \begin{pmatrix} \sigma^2_{max} & \rho_{max,SD} \\ \rho^Z_{max,SD} & \Sigma_{SD} \end{pmatrix} \end{aligned} \tag{2.32}$$

Then the conditional PDF $p_c(Z_{max}|t_s, Z_D)$ is a Gaussian distribution [29], and its mean $\widehat{\mu}_{max}$ and variance $\widehat{\sigma}_{max}$ are given by

$$\widehat{\mu}_{max} = \mu_{max} + \rho_{max,SD}\Sigma^{-1}_{SD}(t_s(Z_D) - \mu_{SD}) \tag{2.33}$$

$$\widehat{\sigma}^2_{max} = \sigma^2_{max} - \rho_{max,SD}\Sigma^{-1}_{SD}\rho^S_{SD} \tag{2.34}$$

where according to equation (2.30)

$$Z_{SD} = \begin{pmatrix} t_s(Z_D) \\ Z_D \end{pmatrix} \tag{2.35}$$

Performing integration of the Gaussian PDF in (2.29),

$$\widehat{\mu}_{max} = t_h - \widehat{\sigma}_{max}\Phi^{-1}(q + \frac{1}{\lambda}) \tag{2.36}$$

where $\Phi(x)$ is the standard normal CDF.

Substituting expressions for $\rho_{max,SD}$, Σ_{SD}, Z_{SD} and μ_{SD} into (2.33) and performing matrix-vector multiplication it can be shown that

$$\widehat{\mu}_{max} = \mu_{max} + \alpha t_s(Z_D) + \beta Z_D - \alpha\mu_S - \beta\mu_D \tag{2.37}$$

where α and β are expressed through variances and covariances of the sensed max noise, chip max noise and $I_{ddq}(Z_D)$.

Excluding $\widehat{\mu}_{max}$ from (2.36) and (2.37), and solving for t_s

$$t_s(Z_D) = \frac{\beta}{\alpha}(t_h + \mu_D - Z_D) + \mu_S - \frac{\mu_{max}}{\alpha} + \frac{\widehat{\sigma}_{max}}{\alpha}\Phi^{-1}(q + \frac{1}{\lambda}). \tag{2.38}$$

It is observed that threshold is a linear function of $I_{ddq}(Z_D)$. Rewriting it as

$$t_s(Z_D) = \gamma Z_D + \eta \tag{2.39}$$

The Lagrange multiplier λ can easily be found by computing η. By changing the order of integration in the numerator of (2.25) and transforming nested integrals into an integral over the area $Z_S \leq t_s(Z_D) = \gamma Z_D + \eta$,

$$\frac{\int_{t_h}^{\infty}(\iint_{Z_S \leq \gamma Z_D + \eta} p_c(Z_{max}, Z_S, Z_D) dZ_S dZ_D) dZ_{max}}{\iint_{Z_S \leq \gamma Z_D + \eta} p_t(Z_S, Z_D) dZ_S dZ_D} = q. \tag{2.40}$$

Rotating the coordinate system by variable transformations

$$Z_D = \frac{u - \gamma v}{\sqrt{1+\gamma^2}}, Z_S = \frac{\gamma u + v}{\sqrt{1+\gamma^2}} \tag{2.41}$$

and converting the integrals over the area back into nested integrals, and getting

$$\frac{\int_{t_h}^{\infty}\int_{-\infty}^{\frac{\eta}{\sqrt{1+\gamma^2}}}\int_{-\infty}^{\infty} p_c(Z_{max}, \frac{\gamma u + v}{\sqrt{1+\gamma^2}}, \frac{u - \gamma v}{\sqrt{1+\gamma^2}}) du dv dZ_{max}}{\int_{-\infty}^{\frac{\eta}{\sqrt{1+\gamma^2}}}\int_{-\infty}^{\infty} p_t(\frac{\gamma u + v}{\sqrt{1+\gamma^2}}, \frac{u - \gamma v}{\sqrt{1+\gamma^2}}) du dv} = q. \tag{2.42}$$

The region of integration of the two inner integrals of the numerator is a half plane, and these integrals can be expressed analytically in terms of the standard Gaussian CDF function $\Phi(x)$. This transforms (2.42) into a single integral.

(2.42) can be efficiently solved for η by any root-finding technique with numerical integration technique. Substituting the computed value of η into (2.39), the optimal value of the soft threshold can be achieved.

2.6.2. Per-Chip Soft Threshold Computation with Varying Hard Threshold. It should be pointed out that the soft threshold voltages set according to the above approach are pessimistic, as it has not considered the fact that the hard threshold t_h may also vary due to process variations. Figure 2.3 shows hard threshold for different I_{ddq} test and suggests a linear relation between I_{ddq} and hard threshold for a NAND gate. The hard threshold t_h is calculated as the maximum voltage drop allowed to keep the delay above a constant threshold. From the figure it can be seen that as I_{ddq} increases, t_h also increases. This is expected as a larger I_{ddq} actually corresponds to a faster gate and accordingly more room is available for delay degradation.

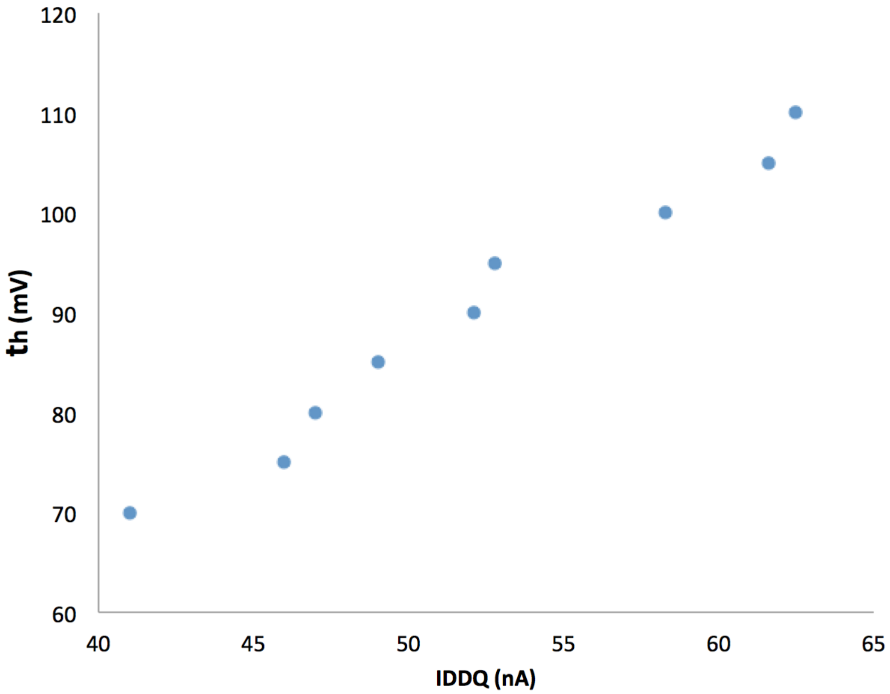


Figure 2.3: Relation between I_{ddq} and hard threshold t_h. The experiment is done by HSPICE simulation of a NAND Gate in 45nm node

Thus, lower I_{ddq} will result in slower chip, and accordingly lower t_h under the same supply voltage and clock frequency. As a result, system failure rate increases with decreased t_h. In other words, the t_s obtained from the framework in Section 2.6.1 will result in system failure rate constraint violation when I_{ddq} gets lower (over pessimistic). This can also be clearly seen from Figure 2.3. Accordingly, it turns to solve the following problem: repeat (2.22) and (2.23) with t_h changed to $t_h(Z_D)$.

Assume there is a linear relationship between I_{ddq} and Z_D, i.e.,

$$t_h = \kappa Z_D \tag{2.43}$$

where κ is some constant that can be decided once the design and process is determined. Thus (2.40) turns to

$$\frac{\int \big(\int_{\kappa Z_D}^{\infty} \int_{Z_S \leq \gamma Z_D + \eta} p_c(Z_{max}, Z_S, Z_D) dZ_S dZ_{max}) dZ_D}{\iint_{Z_S \leq \gamma Z_D + \eta} p_t(Z_S, Z_D\big) dZ_S dZ_D} = q. \tag{2.44}$$

After solving (2.44), to get η, which has to be solved numerically in this case, the optimal soft threshold for varying hard threshold with (2.39) can be achieved. In (2.44), denote

$$T = \frac{\int \big(\int_{\kappa Z_D}^{\infty} \int_{Z_S \leq \gamma Z_D + \eta} p_c(Z_{max}, Z_S, Z_D) dZ_S dZ_{max}) dZ_D}{\iint_{Z_S \leq \gamma Z_D + \eta} p_t(Z_S, Z_D\big) dZ_S dZ_D} \tag{2.45}$$

thus

$$\frac{dT}{d\kappa} = \frac{-Z_D(\iint_{Z_S \leq \gamma Z_D + \eta} p_c(Z_{max}, Z_S, Z_D) dZ_S dZ_D) dZ_{max}}{\iint_{Z_S \leq \gamma Z_D + \eta} p_t(Z_S, Z_D) dZ_S dZ_D} < 0 \tag{2.46}$$

(2.46) suggests the higher the κ, the smaller the system failure rate. According to Figure 2.1, a less strict (higher) t_s can be computed under the constraint.

From the above derivation, it is clear that it the optimal soft threshold for each chip based on I_{ddq} measurement can be determined.

It is worthwhile to justify the feasibility of our per-chip soft threshold setting approach. This approach, although novel in the context of noise sensors, is tightly linked to the established practice of per-chip supply voltage setting based on process variations [6, 36]. If the threshold voltage for the noise sensors comes from external reference, then it can easily tuned by adjusting the corresponding voltage regulator on the board. However, such a practice can be expensive for large volume productions. To save cost, we can adopt a voltage binning based approach[19, 21] by implementing several voltage regulators with different output voltages on chip and using a MUX to select the one nearest to the computed optimal threshold. This will result in higher alarm rate under the same system failure rate constraint, but can reduce the cost drastically.

In terms of time complexity, three steps are needed for fixed hard threshold and varying hard threshold optimal soft threshold computation as follows: 1) Measure I_{ddq} of each chip during testing; 2) Calculate the optimal soft threshold t_s following Section 2.6; and 3) Set the soft threshold t_s for each chip. Step 1 is standard in chip testing, so no extra cost is incurred. Step 2 takes little time as only a single variable equation needs to be solved. Step 3 bears a cost similar to existing practices of per-chip supply voltage setting, so is also applicable to large volume production.

2.7. EXPERIMENTAL RESULTS

Various optimal threshold voltage computation methods are implemented discussed above in MATLAB, and performed experiments on a workstation with dual six-core, 2.4 GHz, Intel Xeon E5645 CPU and 96 GB memory. A set of three power grids extracted from in-house designs at 45 nm technology node are used, with detailed info listed in Table 2.1.The noise is obtained using SPICE simulation and modeled using the approach discussed in Section 2.4. For each design, 10 sensors are placed according to the method described in [41]. The threshold time T_t is set to 1 ns for *intuitive*, *exact* and *per-chip* methods based on nominal design. Finally, to include process variation impact, for each design 40 chips are simulated with different process parameters sampled according to the foundry rule. The hard threshold for the intuitive method, the exact uniform method and the per-chip method with fixed hard threshold is set based on the nominal case. Design N1 is first used to compute the intuitive soft threshold from Section 2.5.1, the exact uniform soft threshold from Section 2.5.2 and the per-chip fixed hard threshold and per-chip varying hard threshold methods from Section 2.6, for different system failure rate requirements.

Table 2.1: Benchmark information. #i, #n, #r stand for the number of current sources, nodes and resistance, respectively.

Design	#i	#n	#r
N1	5,387	5,387	4,720
N2	18,419	19,240	38,366
N3	100,527	102,178	197,470

Figure 2.4 shows the comparison of achieved system failure rate between different thresholds at various required system failure rates, averaged over 40 chips. As expected, the system failure rate achieved by the per-chip method with varying hard threshold is almost the same as the required system failure rate, higher than those achieved by intuitive and exact methods. When system failure rate constraint is tight, achieved system failure rate for per-chip method with fixed threshold may cause system failure rate violation, because it does not consider the possible variation in t_h. While the intuitive method and the uniform method also assume fixed hard threshold, the large margin between the achieved system failure rate and the required system failure rate prevent the violation from happening. The intuitive uniform threshold always gives the lowest system failure rate. This suggests that among the three thresholds, only the per-chip threshold can fully utilize the allowed system failure rate to attain the lowest system failure rate by taking the most detailed chip information into consideration. When the required system failure rate is loose enough (e.g. 5%), all the three thresholds from intuitive, exact and per-chip methods approach the hard threshold in the nominal case, and accordingly they all achieve the same system failure rate on average.

To verify the above discussion, same design and setting are used to compare the alarm rate rate between the three thresholds for different system failure rate requirement, and the results are depicted in Figure 2.5. Apparently, the per-chip threshold achieves the lowest rate among the three. Per-chip method with fixed and varying hard threshold can reduce the alarm rate by 21.7% and 12.3% on average compared with uniform threshold approach. Alarm rate of per-chip method with fixed hard threshold is the lowest because it does not consider the possible variation in t_h and has potential system failure rate violations.

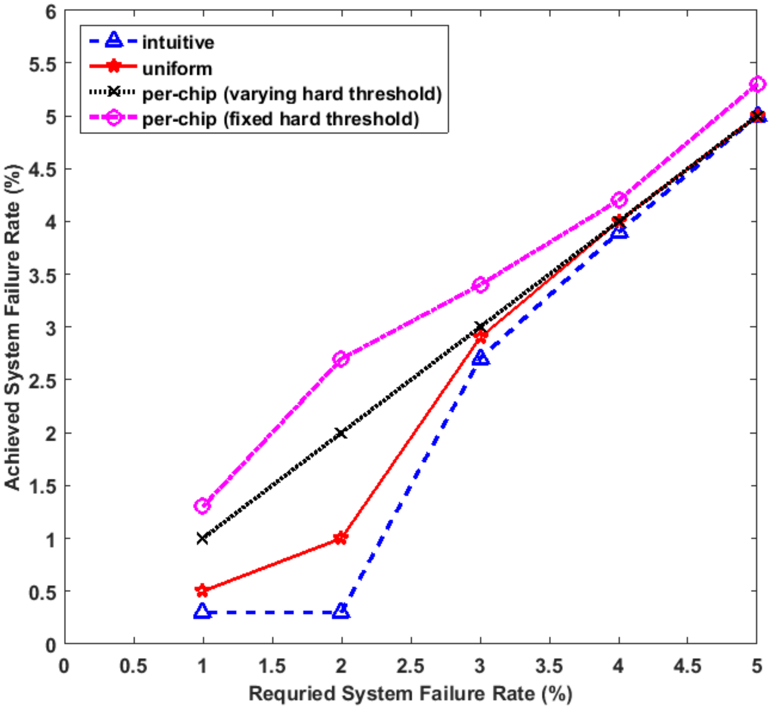


Figure 2.4: Achieved system failure rate vs. required system failure rate for different soft threshold computation methods using design N1

Figure 2.6 and Figure 2.7 show the computed threshold under different system failure rate constraints. Not surprisingly, the intuitive threshold is the highest and the per-chip threshold is the lowest, which agrees with the observations in Figure 2.4 and Figure 2.5. When required system failure rate is high enough, all the four thresholds are equal to the hard threshold. In addition, as shown in Section 2.6, optimal per-chip

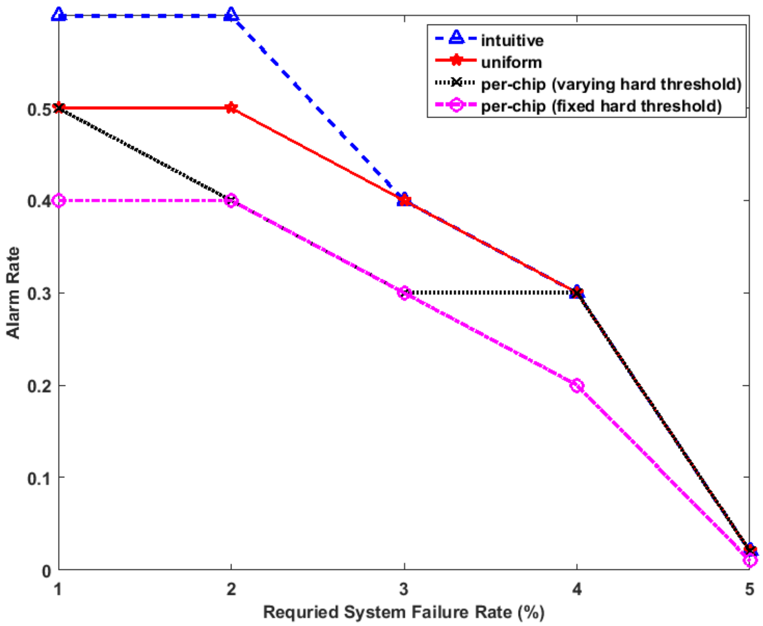


Figure 2.5: Alarm rate vs. required system failure rate for different soft threshold computation methods using design N1

threshold is a linear function of the measured I_{ddq}, which is also verified in Figure 2.7. As chip noise increases due to process variation (i.e., measured I_{ddq} increases), a higher soft threshold is needed for the same system failure rate value. Figure 2.8 shows the linear relation between computed soft threshold and κ. In different technology node, κ is different, thus resulted in different soft threshold t_s.

Finally, Table 2.2 gives the complete comparison results on achieved system failure rate and alarm rate for all the designs, under different required system failure rate. From the table it can be seen that the per-chip threshold always gives the highest achieved system failure rate, and accordingly the lowest alarm rate. It is worthwhile to note that for N3, the system failure rate achieved by per-chip threshold is sometimes smaller than the required system failure rate. This is primarily due to the modeling error (e.g. in Gaussian approximation and PCA) when the design is large. On average, the exact uniform threshold achieves 20.6% lower system failure rate compared with the intuitive uniform threshold under the same system failure rate

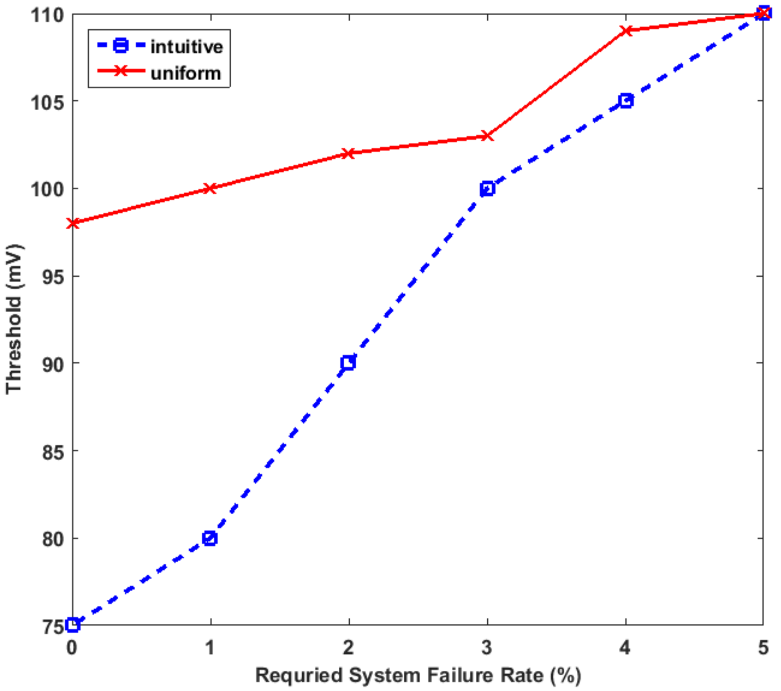


Figure 2.6: Intuitive and exact uniform thresholds vs. required system failure rate using design N1

constraint. In addition, the per-chip threshold is able to further reduce the system failure rate by 12.3% compared with the exact uniform threshold.

2.8. CONCLUSIONS

In this chapter, the problem of optimal soft threshold voltage computation is formally formulated and solved, which is important for runtime noise management systems. Compared with an intuitive approach which tends to be too conservative, an exact approach is first proposed and achieves an average of 20.6% reduction in alarm rate under the same system failure rate constraint. Furthermore, by utilizing the I_{ddq} information which can be easily measured during testing, the uncertainties from process variations can be partially captured, leading to per-chip optimal approach further reduce the system failure rate by 12.3% on average compared with uniform threshold approach.

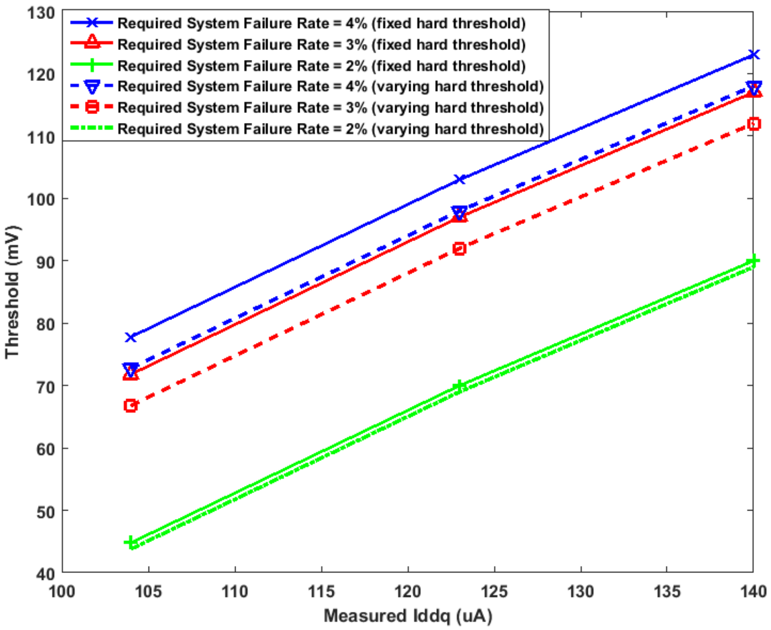


Figure 2.7: Per-chip threshold vs. measured I_{ddq} at different required system failure rate using design N1

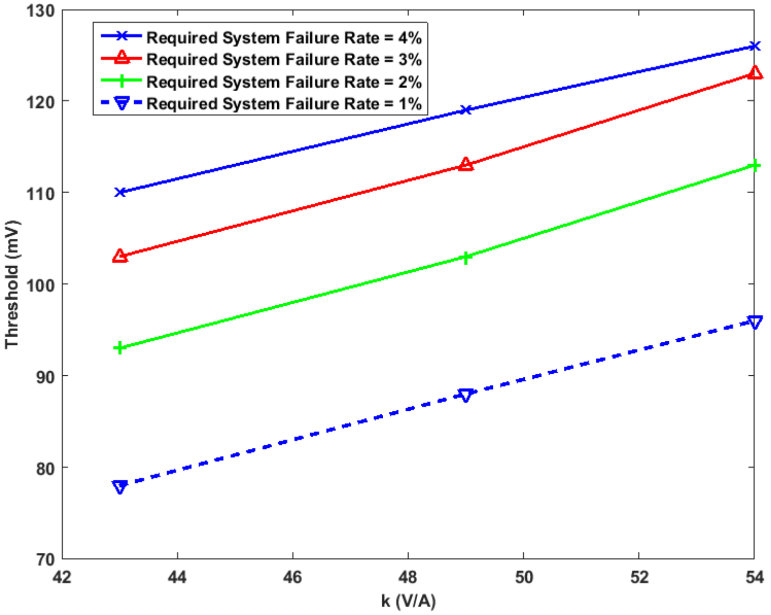


Figure 2.8: Per-chip (varying hard threshold) threshold vs. κ at different required system failure rate using design N1

Table 2.2: Comparison of different soft threshold computation methods

Benchmark	Required system failure rate		1%	2%	3%	4%	5%
N1	Achieved system failure rate	Intuitive	0.3%	0.3%	2.7%	3.9%	5%
		Exact	0.5%	1.0%	2.9%	4%	5%
		Per-Chip (fixed hard threshold)	1.3%	2.7%	3.4%	4.2%	5.3%
		Per-Chip (varying hard threshold)	1.0%	2%	3%	4%	5%
	Alaram Rate	Intuitive	0.6	0.6	0.4	0.3	0.02
		Exact	0.5	0.5	0.4	0.3	0.02
		Per-Chip (fixed hard threshold)	0.4	0.4	0.3	0.2	0.01
		Per-Chip (varying hard threshold)	0.5	0.4	0.3	0.3	0.02
N2	Achieved system failure rate	Intuitive	0.5%	0.8%	1.0%	2.0%	3.0%
		Exact	0.5%	0.8%	2.0%	2.0%	3.4%
		Per-Chip (fixed hard threshold)	1.3%	2.1%	3.0%	4.3%	5.7%
		Per-Chip (varying hard threshold)	1.0%	2.0%	3.0%	4.0%	5.0%
	Alarm Rate	Intuitive	0.4	0.3	0.3	0.1	0.09
		Exact	0.4	0.3	0.1	0.09	0.09
		Per-Chip (fixed hard threshold)	0.3	0.3	0.1	0.04	0.03
		Per-Chip (varying hard threshold)	0.3	0.3	0.1	0.08	0.04
N3	Achieved system failure rate	Intuitive	1.0%	1.0%	2.0%	2.0%	2.0%
		Exact	1.0%	2.0%	3.0%	3.0%	3.0%
		Per-Chip (fixed hard threshold)	1.0%	2.0%	3.0%	5.0%	5.0%
		Per-Chip (varying hard threshold)	1.0%	2.0%	3.0%	3.0%	5.0%
	Alarm Rate	Intuitive	0.3	0.3	0.3	0.2	0.2
		Exact	0.3	0.2	0.1	0.1	0.1
		Per-Chip (fixed hard threshold)	0.3	0.2	0.1	0.03	0.03
		Per-Chip (varying hard threshold)	0.3	0.2	0.1	0.1	0.03

3. 1-BIT COMPRESSED SENSING BASED FRAMEWORK FOR BUILT-IN RESONANCE FREQUENCY PREDICTION USING ON-CHIP NOISE SENSORS

3.1. ABSTRACT

Significant noise will occur when the load currents of a chip contain frequency components that are close to its resonance frequency, which is mainly decided by power delivery network (PDN) capacitance and package inductance. Yet with technology scaling, the wire parasitic capacitance, which suffers from large process variations, starts to become a dominant contributor in the PDN capacitance, leading to a large resonance frequency variation across dies. It is thus important to know the resonance frequency of individual chips to effectively avoid resonance noise at runtime. Existing methods are mostly based on frequency sweeping, which are too expensive to apply to individual chips. In this chapter, a novel framework to predict the resonance frequency is proposed using existing on-chip noise sensors, based on the theory of 1-bit compressed sensing. Experimental results on industrial designs show that compared with frequency sweeping, the proposed framework can achieve up to 7.6× measurement time reduction under the same accuracy, with 15% resonance frequency variation. the need of as well as a practical solution to the resonance frequency prediction for individual chips.

3.2. INTRODUCTION

The capacitance and inductance components in the power delivery network (PDN) from board, package and die form resonant tanks that resonate at multiple frequencies. The dominant resonance frequency f_{res}, mainly due to the package inductance and on-die PDN capacitance [39], usually occurs at low-to middle frequency range (50 MHz to 400 MHz) [42]. Figure 3.1 shows an example PDN impedance plot from an industrial design. As can be seen from the figure, the peak at f_{res} is much sharper and higher than at other resonance frequencies.

When the current spectrum of a chip contains a frequency component close to f_{res} (e.g., due to looping sequences of instruction execution), it will introduce persistent undershoots and overshoots known as *first droop* or *resonance noise*, a major focus of power integrity engineers. Resonance noise not only compromises chip performance and hold-time margins, but also impairs gate oxide integrity or even causes chip breakdown [34] [44]. Due to the very nature of PDN, resonance noise cannot be fully removed. It is therefore a crucial design target to control the impact of resonance noise.

Unlike IR drop noise which can be suppressed through design time solutions such as decoupling capacitance (decap) insertion [35] or wire sizing [8], the reduction of resonance noise largely relies on runtime schemes to prevent load spectrum from hitting the resonance frequency. For example, a frequency actuator based method is proposed in [34], while the authors of [43] provide an on-die resonance-suppression circuit technique that uses band-limited active damping. In [28], load patterns that lead to resonance frequency are pre-characterized and scrambled/manipulated upon detection at runtime. A critical requirement for all these approaches is the *a-priori* information of the resonance frequency.

Conventionally, the resonance frequency is obtained through frequency sweeping. For example, the methods in [17][18] use a specially designed set of instructions to create two known running states and measure the PDN impedance at various frequencies. On the other hand, the methods in [39][42] use clock switching to achieve similar target. All these methods require one measurement at each frequency point. The resonance frequency can then be obtained by finding the frequency that corresponds to the maximum impedance. Apparently, the impedance sweeping is a very time-consuming task and can only accommodate a limited number of test/sample chips. Accordingly, these methods are effective only when all the chips fabricated for

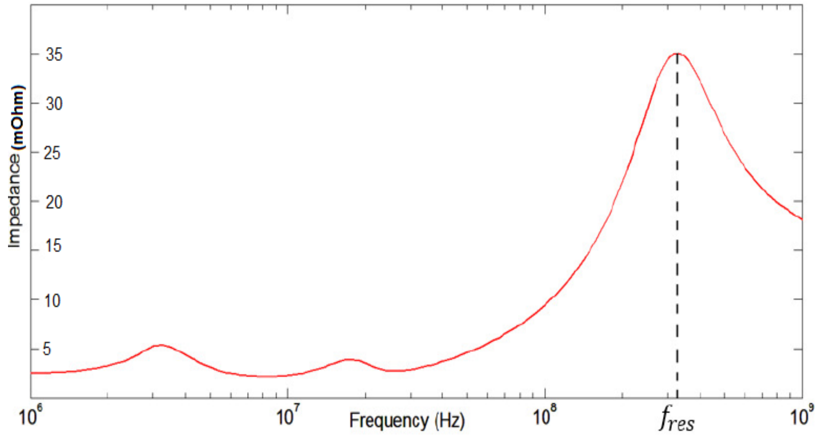


Figure 3.1: A representative PDN impedance from an in-house 45 nm industrial design

the same design have very similar resonance frequencies. This assumption has been taken for granted for many years.

Unfortunately, with details to be discussed in Section 3.3.2, it is no longer valid as technology scales beyond 45 nm. The main cause is the gradual dominance of wire capacitance, which has large variations, in on-die PDN capacitance. It is thus imperative to have an efficient method to estimate the resonance frequency of each individual chip during testing. Unfortunately, this problem has never been paid attention to in the literature, not to mention the corresponding solutions.

In this chapter, a novel built-in resonance frequency prediction framework is proposed for per-chip application. It takes advantage of on-chip noise sensors, which are placed for runtime voltage emergency detection, to measure the noise for different load patterns. It then estimates the resonance frequency using 1-bit compressed sensing theory, a recent advance in signal processing. Experimental results on a few 45 nm industrial designs show that compared with frequency sweeping, our proposed framework can achieve up to 7.6× measurement time reduction under the same accuracy, with 15% resonance frequency variation. To the best of the authors knowledge, this is the very first work to present the need for per-chip resonance frequency prediction, and the first to put forward a practical solution.

The remainder of this chapter is organized as follows. The impact of resonance frequency variation and the design of on-chip noise sensors are reviewed in Section 3.3. The per-chip resonance frequency prediction framework is presented in Section 3.4. Experimental results are discussed in Section 3.5 and concluding remarks are given in Section 3.6.

3.3. PRELIMINARIES AND MOTIVATION

3.3.1. On-Die Capacitance Breakdown and Its Impact on Power Supply Noise. On-die decaps between a specific power domain and the ground net includes the *intentional* decap inserted by designers and the *intrinsic decap*. Intentional decap, like metal-oxide-semiconductor (MOS) decap and metal-insulator-metal (MIM) cap, is known to the designers and can be counted from the layout. In tradition, the major contributors to the intrinsic decap are considered to be wire capacitance and active devices capacitance [45]. Wire capacitance includes power/ground and signal wires coupling capacitance, which plays an important role in the intrinsic decap value. As the technology enters into sub-22nm regime, due to the selective scaling which only reduces wire length and width but not height to avoid resistance increase, wire capacitance is gradually dominated by fringing capacitance. Compared with decap (gate capacitance) which scales with both gate length and width, fringing capacitance only scales with wire length, which is much slower. Thus, in many IPs, it is common that the wire cap is more than the device parasitics cap, which has been silicon validated on on-market products [45]. For example, the on-die capacitance breakdown of an in house 32 nm DDR I/O test chip is shown in Table 3.1. From the table it is clear that the wire capacitance constitutes more than half of the total on-die capacitance.

Table 3.1: On-die capacitance breakdown of a 32 nm DDR I/O test chip

Intrinsic decap				Intentional decap	Total
PG coupling	Active device	Power-to-signal	Ground-to-signal		
17%	3%	26%	4%	49%	99%

Furthermore, driven by cost saving, in some power domains, substantial amount of intentional decap is removed and the system relies on the intrinsic cap to maintain functionality. While for some sensitive domains, like PLL, intentional decap is still dominant. For interconnect dominant domains, like I/O, DDR controller, due to the low cost guideline and area scaling demands, the intentional decap is kept at a minimum value. The system relies on the parasitics cap to ensure operation. For example, it has been validated that even after removing all the intentional decaps in the DDR I/O design, the chip can still function correctly.

As a result, wire capacitance will gradually become dominant in PDN capacitance for many IPs, esp. the I/Os for low cost SoC chips. This can be clearly seen in Figure 3.2 obtained from [45][1]. The recent trends of runtime noise management system deployment [33][11] and silicon cost reduction [2] further reduce decap and speed up the domination of wire capacitance.

3.3.2. Resonance Frequency Variation. For 45 nm technology and beyond, the variation of resonance frequency increases drastically with technology scaling, which can be inferred from the following two observations.

First, as mentioned in Section 3.3.1, wire capacitance will dominate decap in the near future. Second, it has been established that due to process variations the total decap in a chip does not vary much [26], while the wire capacitance can vary greatly (from 10%-25%) [13]. The large variation in wire capacitance comes from the fact that PDN wires are very thick and long and any small variation in wire geometry will lead to a much larger variation in its capacitance. Specifically, the combination of all process variations (lithography, etch, chemical-mechanical planarization) results

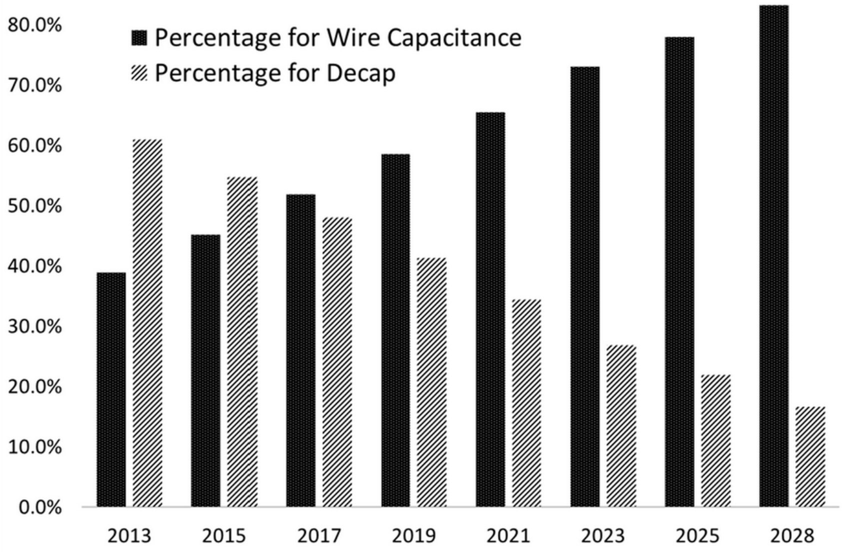


Figure 3.2: Percentage of decap and wire capacitance in on-die PDN capacitance [1]. Wire capacitance will dominate in the near future

in variations of wire spacing, height, profile and metal composition (barrier/copper ratio), which in turn strongly influence wire capacitance values. Such variation will further increase with technology scaling.

Combining the above two observations, it is now obvious that the variation in on-die PDN capacitance, and accordingly the variation in resonance frequency, will increase with technology scaling. To see the impact of such variation on noise, Figure 3.3 shows the maximum noise in a PDN loaded with periodic ON/OFF switching currents. The frequency of the current load is changed from 10 Hz to 10 GHz and the resonance frequency f_{res} of this PDN is 100 MHz. It can be observed from the figure that the resonance noise increases sharply when the load frequency approaches f_{res}. Also, multiple local maxima can be observed when f_{res} is some (small) multiples of the load frequency. The maxima decrease as they move away from f_{res}. All these suggest that a small deviation in resonance frequency prediction can lead to a large increase in the noise by letting the load frequency or its multiples get closer to f_{res}.

3.3.3. On-Chip Noise Sensors. While many different types of noise sensor designs exist in literature, the On-Die Droop Detector design in [23] is chosen to be

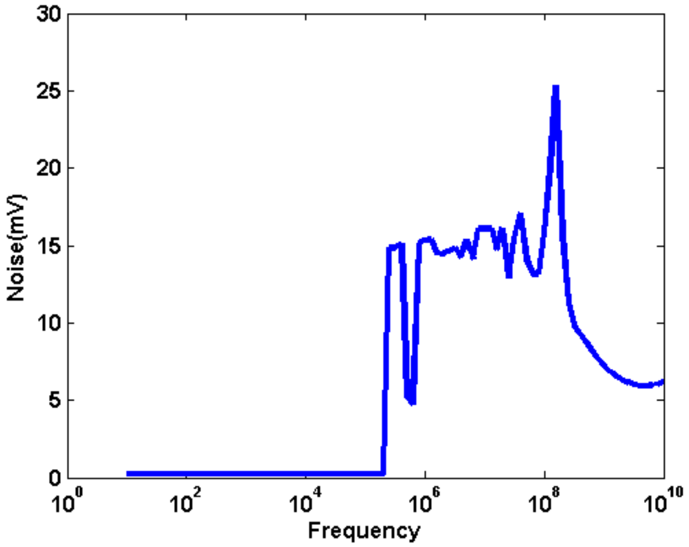


Figure 3.3: Resonance noise in PDN for load patterns with different frequencies

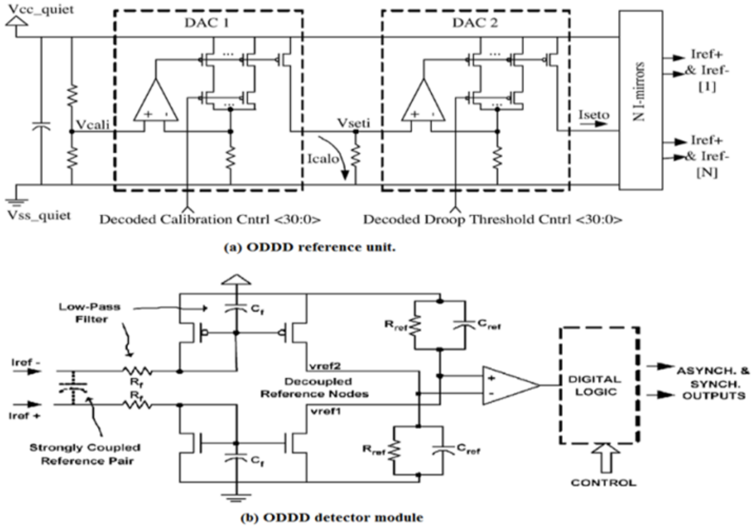
reviewed as it has been widely applied in both academia and industry. However, it is understood that the framework to be proposed later can be applied to any noise sensor designs performing the same duty.

The detailed design is shown in Figure 3.4, which is composed of the Reference Unit and the Detector Module(s). The Detector Module(s) utilize the reference currents Iref+/Iref-, provided by the Reference Unit, to set the differential threshold voltage V_{ref}. The binary output of the Detector Modules indicates if the noise has exceeded the threshold.

Such a binary output is common in most noise sensor designs. Instead of directly providing analog noise amplitudes, they provide a binary output indicating whether the noise has surpassed a threshold voltage or not to support control decisions such as throttling or rollback. This is to enhance the reliability of the design and more importantly, to reduce the associated hardware cost. Only a single wire is needed to connect the sensor to the controller.

In this chapter, it is assumed that all the noise sensors are placed (e.g., following the method in [40] so that the maximum noise in a chip can always be captured).

In addition, it is assumed that the threshold of these sensors is configurable during testing stage - only one threshold voltage is sufficient for this chapter's purpose.



(a) ODDD reference unit.

(b) ODDD detector module

Figure 3.4: On-Die Droop Detector (ODDD) proposed in [23]

3.4. PROPOSED FRAMEWORK

In this section, the overall framework of the per-chip resonance frequency prediction system is first presented, and then each module is discussed in detail.

3.4.1. Overview. The proposed general approach is based on the recovery of PDN impedance through measured noise. This seems to be similar to the existing impedance characterization methods [17][18][39][42]. However, the author aim at developing a built-in framework (i.e. no external measurements) and avoid the costly frequency sweeping. Accordingly, three major differences, all for the sake of prediction efficiency improvement, push us to seek for a new solution.

First, instead of measuring noise externally, on-chip noise sensors which can significantly save time and measurement cost are used. It is worthwhile to note that these sensors are placed for runtime noise management, and the proposed framework takes advantage of them for an alternative purpose. As a tradeoff, it will not be able to get the amplitude of the noise, but rather a binary signal indicating whether the noise has exceeded a pre-set threshold (details in Section 3.3.3). This poses a challenge in the impedance recovery.

Second, since all the loads must be generated on chip, it is not possible to have single-frequency loads (i.e., sine waveforms). The only shape available is the periodic triangular waveform, which is widely used to model short-circuit currents during switching [34][35]. The waveform contains the fundamental frequency decided by the period as well as its multiples. This in fact leads to both opportunity and challenge in the impedance recovery.

Finally, it is assumed that the impedance can be treated as sparse in the frequency domain in the sense that the amplitude at resonance frequency (peak) is much higher than those at frequencies far away from it. This brings an opportunity for efficiency improvement, through the utilization of various compressed sensing frameworks.

The proposed hardware framework for per-chip resonance frequency prediction is shown in Figure 3.5.

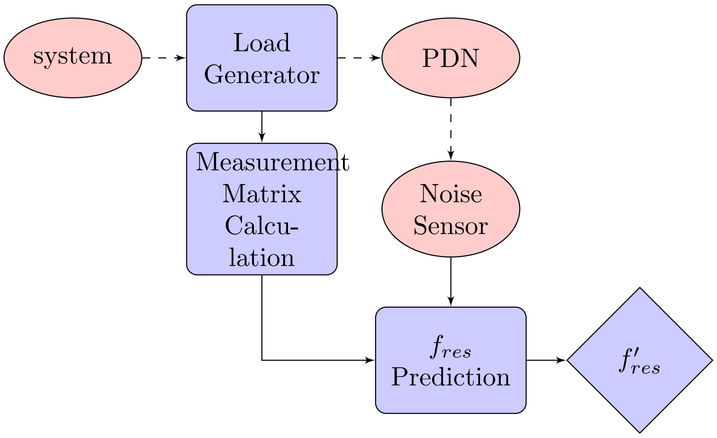


Figure 3.5: Per-chip resonance frequency prediction framework

Load Generator module generates a low frequency clock signal BCLK which bypasses regular clock signal and drives flip-flops across the chip for noise generation.

Measurement Matrix Calculation module calculates the amplitudes of the harmonics for a few loads driven by BCLK signals of different frequencies and provides the measurement matrix needed for impedance recovery. The noise sensors are applied to compare the generated noise with the pre-set threshold.

Finally, using the binary outputs from the noise sensors and the related measurement matrix, the *Resonance Frequency Prediction* module predicts the resonance frequency and reports the result.

The details of the Load generator will be provided, Measurement Matrix Calculation and Resonance Frequency Prediction modules below.

3.4.2. Load Generator. Load generation scheme as shown in Figure 3.6 is proposed to use, which generates a periodic current consumption stimulated at variable low frequencies in the bypass mode. When BYPASSEN signal is low, regular clock generated by PLL will be driven to clock node CLK through the MUX and the clock driver CLKDRV. When BYPASSEN is high, the bypass clock signal BCLK, generated by a T flip-flop driven by a programmable counter, will be driven to the CLK node. CLK can drive the flip-flops across the chip via CLK network, which can create a very representative stimulus that consumes the current from all power delivery bumps of the die.

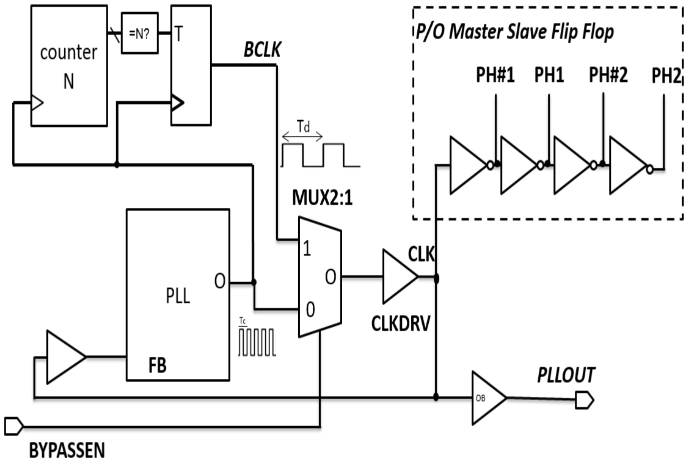


Figure 3.6: Load generation scheme

Two things are worthwhile noting here. First, the counter and the T flip-flop are driven by the clock from PLL, so the period of the output of the T flip-flop T_d must be some multiples of the period of the clock from PLL T_c. Accordingly, the period of the load currents to the PDN will also be T_d. By programming the counter, T_d can be changed.

Second, although BCLK is a square waveform which contains frequency components mainly at DC, f_d and $3f_d$ ($f_d = 1/T_d$), the load currents are periodic triangular-shaped waveforms. As the rise and fall time of the triangles are very small (a few ps at most), the spectrum of the load currents has abundant discrete components in a wide frequency range. To see this, in Figure 3.7 the spectrum of a single triangular waveform with 10 ps rise and fall time has been plotted. As can be seen from the figure, significant frequency components exist up to 60-80 GHz. Depending on the BCLK frequency f_d, the spectrum of the corresponding periodic triangular waveforms will just be the discrete sampling of the one shown in Figure 3.7 with f_d as the sampling frequency.

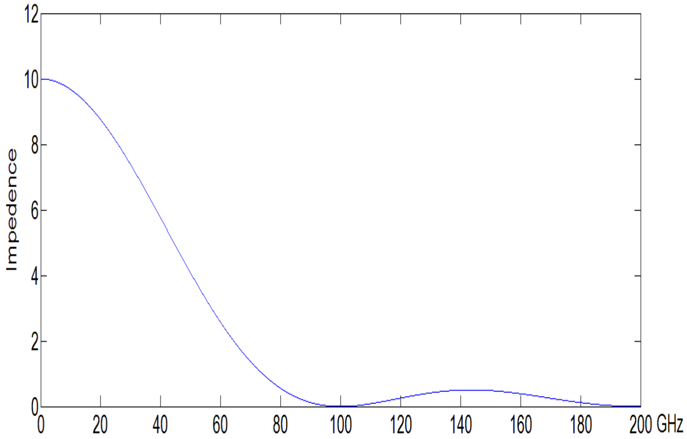


Figure 3.7: Spectrum of a single triangular waveform with 10 ps rise and fall time

3.4.3. Measurement Matrix Calculation. This is the module to generate the matrix needed for impedance recovery. Starting with the Fourier transform of the PDN load $f(t)$ generated by the flip-flops across the chip. As discussed in the previous section, $f(t)$ has a frequency of $f_d = 1/T_d$. The transform can be expressed as

$$F(j\omega) = I_0 \sum_{k=-\infty}^{\infty} a_k \delta(\omega - 2k\pi f_d) \tag{3.1}$$

where I_0 is a constant that reflects the amplitude of the load. a_k are the corresponding Fourier transform coefficients for the load with unit amplitude, which can be obtained directly for any given T_c and T_d. According to the discussion in the previous section, $F(j\omega)$ contains abundant frequency components and a_k will not decay fast with k.

Further denote x as a vector formed by the impedance of the PDN uniformly sampled from $-f_{max}$ to f_{max}, i.e.,

$$x = (x_{-n}, \ldots, x_{-1}, x_0, x_1, \ldots, x_n)^T, \tag{3.2}$$

where x_i is the impedance at $i \times f_{max}/n$. x_0 is the DC impedance. f_{max} needs to be large enough such that the corresponding impedance is much smaller compared with the peak. Apparently, the frequency resolution of such a representation is f_{max}/n.

The PDN voltage response (noise) in the frequency domain can then be expressed as

$$\Delta v(j\omega) = I_0 \sum_{k=-f_{max}/f_d}^{f_{max}/f_d} a_k \delta(\omega - 2k\pi f_d) x_{\frac{kf_d n}{f_{max}}}. \tag{3.3}$$

In the above equation it has implicitly assumed that f_{max}/n is small enough such that f_d is some multiples of f_{max}/n and the corresponding terms in x exist.

As such, the corresponding time domain noise can be obtained by performing inverse Fourier Transform on $\Delta v(j\omega)$ as

$$\Delta v(t) = \frac{I_0}{2\pi} \sum_{k=-f_{max}/f_d}^{f_{max}/f_d} a_k e^{j(2k\pi f_d)t} x_{\frac{kf_d n}{f_{max}}} \tag{3.4}$$

Since the PDN is linear and time-invariant and $f(t)$ is periodic with period T_d, $g(t)$ should also be periodic with the same period. At the end of each period,

$$\Delta v(T_d) = \frac{I_0}{2\pi} \sum_{k=-f_{max}/f_d}^{f_{max}/f_d} a_k e^{j(2k\pi f_d)T_d} x_{\frac{kf_d n}{f_{max}}} = \frac{I_0}{2\pi} \sum_{k=-f_{max}/f_d}^{f_{max}/f_d} a_k x_{\frac{kf_d n}{f_{max}}}. \tag{3.5}$$

(3.5) establishes a linear equation between the impedance vector x and the measured noise in time domain $\Delta v(T_d)$. For simplicity of presentation, superscript is used to denote different loads (measurements). Consider a total of m different loads (m measurements) with frequencies f_d^1, ..., f_d^m. In order to cover all the frequencies in x, it is required that one of these frequencies (assuming f_d^1 without loss of generosity) equals f_{max}/n. This ensures that every element in x will have corresponding coefficients in Φ. It further required that f_{max} is always some multiples of f_d^i ($1 \leq i \leq m$), i.e., $f_{max} = N^i f_d^i$ where N^i is an integer. Then the results can be expressed in a compact matrix form as

$$\Phi x = \Delta V, \tag{3.6}$$

where Φ ($m \times n$) takes the form

$$\Phi = \begin{pmatrix} a^1_{-N^1} & a^1_{-N^1+1} & \cdots & \cdots & \cdots & \cdots & \cdots \\ a^1_{-N^1} & 0 & \cdots & a^1_{-N^1+1}\cdots & \cdots & \cdots & \\ a^2_{-N^2} & 0 & \cdots & \cdots & \cdots & a^2_{-N^2+1} & \cdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \end{pmatrix}$$

When forming Φ, it ignored the constant $\frac{I_0}{2\pi}$ in (3.5), as it only cares the relative amplitudes of different elements in x to find out the resonance frequency. In addition,

$$\Delta V = \begin{pmatrix} \Delta v^1(T_d^1) & \Delta v^2(T_d^2) & \ldots \end{pmatrix}^T \tag{3.7}$$

3.4.4. Resonance Frequency Prediction. Now it is ready to recover x based on (3.6) and the measurement from noise sensors at the end of a cycle ($t = T_d$). In reality the sensors need to wait for the noise to become stable with periodic pattern and then perform the measurement. As mentioned earlier, the sensors will not be able to directly obtain ΔV. Instead, they will return a binary vector y, which indicates if the noise has exceeded a given threshold t. Thus the following equation can be achieved:

$$y = \text{sign}(\Delta V - t) = \text{sign}(\Phi x - t) \tag{3.8}$$

where t is the given threshold. The objective now is to accurately estimate x and find out the largest element.

An efficient solution lies in the fact that, as stated in the beginning of this section, most of the elements in x are very small as compared with the maximum. This inspires us to assume that x can be well approximated by the K most significant coefficients, where K can be decided by the width at the half peak of the impedance. that is, x is K-compressible. Such reconstruction problem can then be solved in compressed sensing frameworks, in which Φ becomes the measurement matrix, and y is the observation vector. This shall enable us to use only a very small number of measurements and recover x accurately. In addition, as the observation vector only contains sign information, recent breakthroughs on 1-bit compressed sensing from the signal processing community [3][14] can be resorted to. Casting the problem as

$$\hat{x} = \arg\min_{x} \|x\|_1 \tag{3.9}$$

$$\text{s.t. } Y(\Phi x - t1) \geq 0 \tag{3.10}$$

where Y is a diagonal matrix and $\text{diag}(Y) = y$.

Pointing out that the above formulation is different from many of those in classic 1-bit compressed sensing framework: there is no constraint that bounds the ℓ_2-norm of x. Such constraint is typically in place to avoid the trivial solution of $x = 0$. However, in this chapter's case, $t > 0$ and thus $x = 0$ is not a feasible solution.

The one-sided ℓ_1-norm in (3.9) is related to the *hinge-loss* function in machine learning literature [12]. This binary classification algorithm seeks to enforce the same consistency function as in (3.10) by minimizing a function $\|\{t1 - y \odot (\Phi x)\}_+\|_1$, where $\{\|u\|_+\}$ sets all negative elements in u to zero and $u \odot v$ denotes the Hadamard product, i.e., $(\mathbf{u} \odot \mathbf{v})_i = u_i v_i$. When $t > 0$, the objective is both convex and has a non-trivial solution.

Thus, rather than minimizing the one-sided ℓ_1 norm, we can minimize the hinge-loss through gradient descent method as

$$a^{l+1} = x^l - \frac{\delta}{2}\Theta^T(\text{sign}(\Theta x^l - t) - 1) \tag{3.11}$$

where $\Theta = y \odot \Phi$ scales the rows of Φ by the signs of y, and δ is a small positive number (0.1 in the experiments). The overview of the proposed algorithm is shown in Algorithm 3.1. The steps are self-evident so this chapter will not explain them in detail in the interest of space.

Algorithm 3.1 Resonance frequency prediction

```
1: INPUT: $y$ from noise sensors, $\Phi$, $f_{max}$ and $n$ from measurement matrix calculation, and $t$, $K$,
   and $\delta > 0$ from user specification.
2: OUTPUT: Estimated $f_{res}$.
3: Set $l = 0$, $x^0 = 0$.
4: Set $a^{l+1} = x^l - \frac{\delta}{2}\Theta^T(A(x^l) - 1)$, where $A(x) = \text{sign}(\Phi x - t1)$;
5: $x^{l+1} = \eta_K(a^{l+1})$, where $\eta_K(u)$ keeps the $K$-terms with maximum absolute values in $u$ and sets
   others to 0;
6: $l = l + 1$.
7: If $d_H(y, A(x^l)) = 0$ or $l = max\_iter$, stop. Otherwise, go to Step 2. $d_H(u, v)$ is the Hamming
   distance between $u$ and $v$.
8: $p = \arg\max_i x_i$; $f_{res} = \frac{p}{n} f_{max}$.
```

3.5. EXPERIMENTAL RESULTS

For rapid prototyping, the resonance frequency prediction framework has been implemented discussed above in MATLAB, and performed experiments on a workstation with dual six-core, 2.4 GHz, Intel Xeon E5645 CPU and 96 GB memory. A set of three on-chip power grid designs extracted from in-house designs at 45 nm technology node are simulated in a commercial SPICE simulator, with detailed info listed in Table 3.2. Their impedance characteristics are characterized through SPICE and reported in Table 3.3. SPICE is also used to simulate the power supply noise under different loads in three benchmarks, which serves as input to the MATLAB framework.

The parasitic inductance and capacitance of the packages are modeled as lumped elements attached to the bumps in the grids. Due to the discrete nature of the proposed method as well as the frequency sweeping based methods, the resonance frequencies are rounded to the nearest integer. As can be seen from the table, the three designs have distinct impedance characteristics, which will play an important role in the results to be discussed. Finally, the load currents are extracted and modeled as triangular waveforms with 10 ps rise and fall time.

The noise sensors are placed according to [40] for each design. It is also assumed that they all use the same threshold in the same design. x is constructed by setting f_{max} to $10f_{res}$ with 1 MHz step. The frequencies of the load patterns f_d are randomly generated in the range between 1 MHz and 100 MHz with 1 MHz resolution. Finally, following Figure 3.2, assuming 15% variation in the resonance frequency. This will decide the number of samples needed by the existing methods to find out the resonance frequency through frequency sweeping [17][18][39][42].

The experiment started with the comparison between the proposed method and the frequency sweeping based methods in terms of the number of measurements needed to get the frequency. The threshold voltage is set to 20 mV. The sweeping based methods need to go through each possible frequency point in the ±15% range of f_{res} with step size 1 MHz. From Table 3.4 it can be seen that the proposed method can achieve up to 7.6× reduction in the number of measurements and thus in measurement time. Note that the speedup increases with the variation in f_{res}. In addition, out of the three designs, N3 needs the fewest measurements while N1 needs the most. This is due to the difference in the impedance peak width of the three designs. As can be seen from Table 3.3, N3 has the narrowest impedance peak and N1 has the widest. A narrower peak corresponds to a sparser vector x, which needs less information to recover. The same table also reports the MATLAB runtime for Algorithm 3.1. Apparently, the runtime is negligible compared with the measurement time, which could be further reduced by hardware implementation.

Table 3.2: Benchmark information. #n, #r, #c, #l stand for the number of nodes, resistance, capacitance and inductance respectively.

Benchmark	#n	#r	#c	#l
N1	30, 638	30,027	10,774	277
N2	127,238	208 325	36,838	330
N3	851,584	1,401,572	201,054	955

Table 3.3: Impedance information for each design. The resonance frequency f_{res}, peak width at half maximum Δf, peak impedance Z_p and DC impedance Z_0 are reported.

Benchmark	f_{res} (MHz)	Δf (MHz)	Z_p (mΩ)	Z_0 ($\mu\Omega$)
N1	158	250	38.1	10.9
N2	80	75	26.7	1.3
N3	126	48	25.8	1.0

Figure 3.8 further demonstrates the frequency prediction error versus the number of measurements for the three designs. The threshold of the sensors is set to 20 mV. From the figure it can be seen that N1 always has the maximum error for the same number of measurements. This is again because N1 has the widest impedance peak. For all the designs, the prediction error drops quickly with the increase of measurement numbers. In addition, Figure 3.9 compares the prediction accuracy versus the threshold of the noise sensors, which is identical for the same design, using 10 measurements. From the figure we can see that an optimal threshold exists for all the three designs. This is because if the threshold is set too low (high), noise sensors will output 1(0) for most of the measurements, resulting in a loss of the information.

Table 3.4: Comparison of the number of measurements needed (#N) between the proposed method and the frequency sweeping based methods to achieve 1 MHz maximum error. The MATLAB runtime of Algorithm 3.1 is also reported.

Ckt	Sweeping based methods	Proposed method	
	#N	#N	runtime (ms)
N1	47 (1)	10 (1/4.7×)	1.5
N2	24 (1)	6 (1/4.0×)	0.9
N3	38 (1)	5 (1/7.6×)	0.7

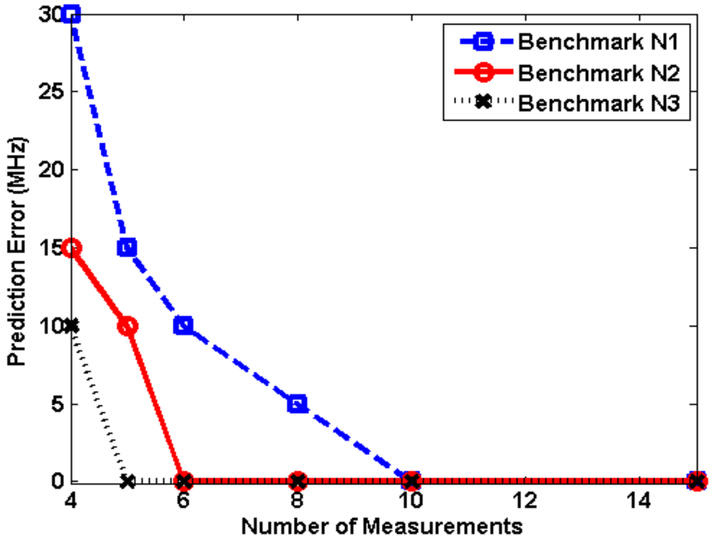


Figure 3.8: Prediction error vs. number of measurements. Sensor threshold = 20 mV

In addition, for the same prediction error, the threshold of N3 is the lowest, while that of N1 is the largest. This is because of the difference in maximum noise (peak impedance), as can be seen from Table 3.3. In future work, how to efficiently select the optimal threshold for a given design will be studied. In the above experiments, it is assumed that any frequency between 1 MHz and 100 MHz can be selected with a minimum resolution of 1 MHz. Figure 3.10 shows prediction error if increasing this minimum resolution (e.g., when the resolution is 2 MHz, only 2 MHz, 4 MHz, 6 MHz, etc. can be selected). A higher resolution results in reduced hardware complexity. From the figure we can see that as expected, for all the three designs the error increases with the resolution. If the prediction error to be within 5 MHz, the resolution cannot be higher than 3 MHz. Figure 3.11 shows the impact of noise in sensor measurements on the prediction accuracy, where SNR is the signal to noise ratio. A zero-mean Gaussian random noise is injected into the voltages measured by the noise sensors. From the figure it can be seen that when SNR > 20 dB, accurate prediction can be guaranteed, which shows that the proposed method has strong resistance against small noise. In addition, N1 is most sensitive to noise (largest slope), followed by N3. N2 is least sensitive. This is due to the difference in impedance peak width. A wider peak needs more information to recover (less sparse) and thus is more sensitive to noise in measurements.

Finally, Figure 3.12 shows the variation of prediction error due to sensor locations for benchmark N1. As it can be observed from Figure 3.12, prediction error is randomly distributed. While the errors are all relatively small (<10MHz) compared to Δf (250MHz), the result suggests that it is possible to find optimal locations of the sensors for best resonance frequency prediction.

3.6. CONCLUSIONS

The increased contribution from wire capacitance due to technology scaling will lead to significant variations in the resonance frequency of PDN, making it imperative to obtain the per-chip information for effective resonance noise management. Unfortunately, existing methods are based on frequency sweeping which is too slow. In this chapter, a novel built-in resonance frequency prediction framework have been proposed using existing on-chip noise sensors. It is based on recent advances in 1-bit compressed sensing. Experimental results on a few industrial designs show that compared with frequency sweeping, the proposed framework can achieve up to 7.6× measurement time reduction under the same accuracy, with 15% resonance frequency variation. To the best of the authors knowledge, this is the very first work to point out the need of as well as a practical solution to the resonance frequency prediction for individual chips.

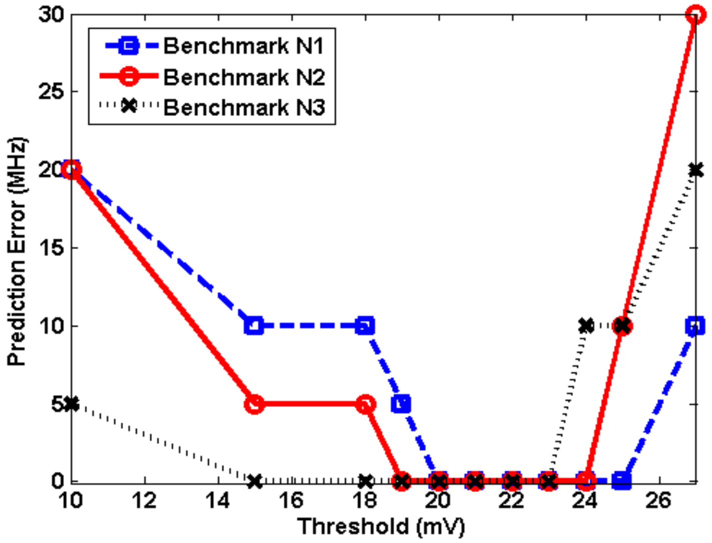


Figure 3.9: Prediction error vs. sensor threshold. Number of measurements = 10

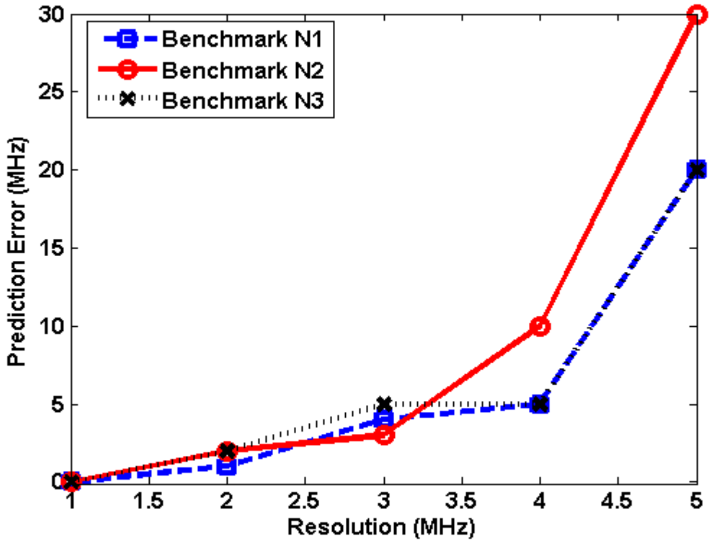


Figure 3.10: Prediction error vs. minimum resolution. Number of measurements = 10, sensor threshold = 20 mV

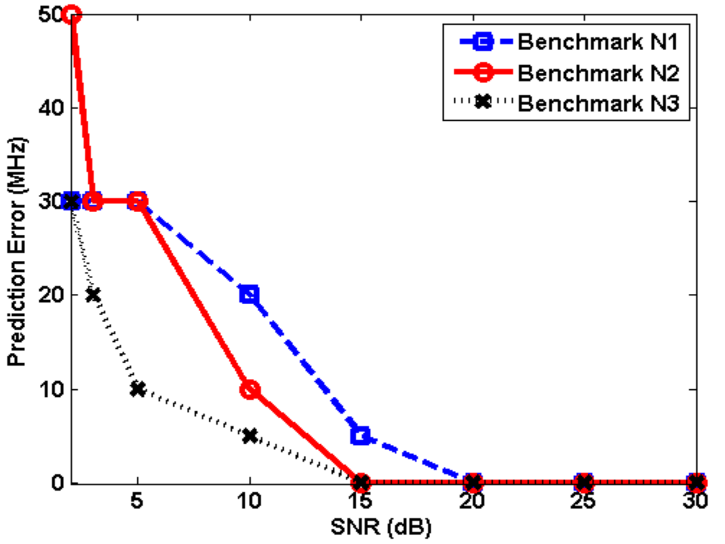


Figure 3.11: Prediction error vs. noise in sensor measurements. Number of measurements = 10, sensor threshold = 20 mV

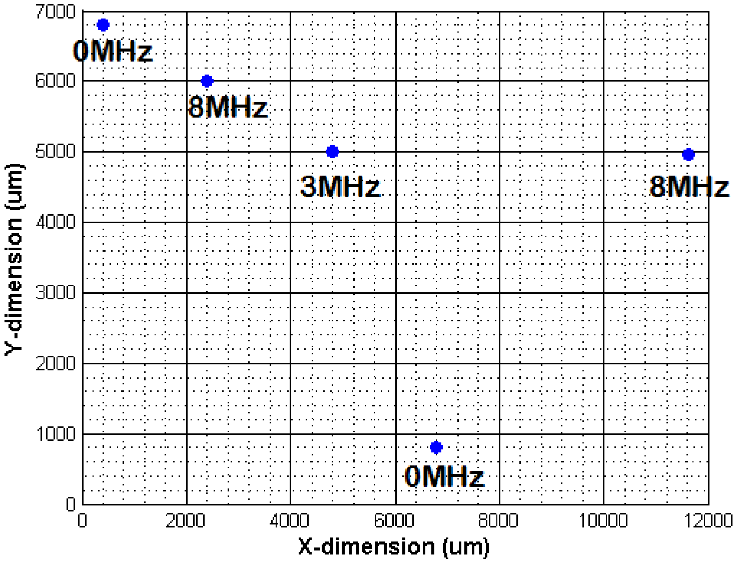


Figure 3.12: Prediction error with sensors in different locations for Benchmark N1. Number of measurements = 10, sensor threshold = 20 mV

BIBLIOGRAPHY

[1] The international technology roadmap for semiconductors. In http://www.itrs.net/. [Online; accessed 19-July-2015].

[2] L. Anghel and M. Nicolaidis. Cost reduction and evaluation of a temporary faults-detecting technique. In *Design, Automation, and Test in Europe*, pages 423–438. Springer, 2008.

[3] P. T. Boufounos and R. G. Baraniuk. 1-bit compressive sensing. In *Information Sciences and Systems, 2008. CISS 2008. 42nd Annual Conference on*, pages 16–21. IEEE, 2008.

[4] B. H. Calhoun and A. P. Chandrakasan. Static noise margin variation for subthreshold sram in 65-nm cmos. *IEEE JOURNAL OF SOLID-STATE CIRCUITS*, 41:1673–1679, 2006.

[5] T. Charania, A. Opal, and M. Sachdev. Analysis and design of on-chip decoupling capacitors. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 21(4):648–658, 2013.

[6] T. Chen and S. Naffziger. Comparison of adaptive body bias (abb) and adaptive supply voltage (asv) for improving delay and leakage under the presence of process variation. *Very Large Scale Integration (VLSI) Systems, IEEE Transactions on*, 11(5):888–899, 2003.

[7] L. Cheng, J. Xiong, and L. He. Non-linear statistical static timing analysis for non-gaussian variation sources. In *44th ACM/IEEE Design Automation Conference*, pages 250–255, 2007.

[8] R. Dutta and M. Marek-Sadowska. Automatic sizing of power/ground networks in vlsi. In *26th ACM/IEEE Design Automation Conference*, pages 783–786, 1989.

[9] T. Enami, S. Ninomiya, and M. Hashimoto. Statistical timing analysis considering spatially and temporally correlated dynamic power supply noise. *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on*, 28(4):541–553, 2009.

[10] U. Feige. A threshold of ln n for approximating set cover. *Journal of the ACM (JACM)*, 45(4):634–652, 1998.

[11] M. S. Gupta, K. K. Rnangan, M. D. Smith, G.-Y. Wei, and D. Brooks. Decor: A delayed commit and rollback mechanism for handling inductive noise in processors. In *IEEE 14th International Symposium on High Performance Computer Architecture*, pages 381–392, 2008.

[12] T. Hastie, R. Tibshirani, J. Friedman, T. Hastie, J. Friedman, and R. Tibshirani. *The elements of statistical learning*, volume 2. Springer, 2009.

[13] V. N. Hoang, A. Kumar, and P. Christie. The impact of back-end-of-line process variations on critical path timing. In *Interconnect Technology Conference, 2006 International*, pages 193–195. IEEE, 2006.

[14] L. Jacques, J. N. Laska, P. T. Boufounos, and R. G. Baraniuk. Robust 1-bit compressive sensing via binary stable embeddings of sparse vectors. *arXiv preprint arXiv:1104.3160*, 2011.

[15] H.-R. Jiang, J.-Y. Jou, and Y.-W. Chang. Noise-constrained performance optimization by simultaneous gate and wire sizing based on lagrangian relaxation. In *36th annual ACM/IEEE Design Automation Conference*, pages 90–95, 1999.

[16] R. Joseph, D. Brooks, and M. Martonosi. Control techniques to eliminate voltage emergencies in high performance processors. In *IEEE 9th International Symposium on High Performance Computer Architecture*, pages 79–90, 2003.

[17] I. Kantorovich, C. Houghton, S. Root, and J. S. Laurent. Measurement of low impedance on chip power supply loop. *IEEE transactions on advanced packaging*, 27(1):10–14, 2004.

[18] I. Kantorovich, C. Houghton, and J. St Laurent. Measurement of milliohms of impedance at hundred mhz on chip power supply loop. In *Electrical Performance of Electronic Packaging, 2002*, pages 319–322. IEEE, 2002.

[19] M. W. Kuemerle, S. K. Lichtensteiger, D. W. Stout, and I. L. Wemple. Integrated circuit design closure method for selective voltage binning, Jan. 6 2009. US Patent 7,475,366.

[20] Z. Li, Y. Zhou, and W. Shi. Wire sizing for non-tree topology. *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on*, 26(5):872–880, 2007.

[21] S. Lichtensteiger and J. Bickford. Using selective voltage binning to maximize yield. In *Advanced Semiconductor Manufacturing Conference (ASMC), 2012 23rd Annual SEMI*, pages 7–10. IEEE, 2012.

[22] N. Mi, J. Fan, S. X.-D. Tan, Y. cai, and X. Hong. Statistical analysis of on-chip power delivery networks considering lognormal leakage current variations with spatial correlation. *IEEE JOURNAL OF SOLID-STATE CIRCUITS*, 41:1673–1679, 2006.

[23] A. Muhtaroglu, G. Taylor, and T. Rahal-Arabi. On-die droop detector for analog sensing of power supply noise. *Solid-State Circuits, IEEE Journal of*, 39(4):651–660, 2004.

[24] A. Muhtaroglu, G. Taylor, and T. Rahal-Arabi. On-die droop detector for analog sensing of power supply noise. *Solid-State Circuits, IEEE Journal of*, 39(4):651–660, 2004.

[25] S. Nassif. Delay variability: sources, impacts and trends. In *IEEE International Solid-State Circuits Conference*, pages 368–369, 2000.

[26] S. R. Nassif, K. Agarwal, and E. Acar. Methods for estimating decoupling capacitance of nonswitching circuit blocks. In *Circuits and Systems, 2006. ISCAS 2006. Proceedings. 2006 IEEE International Symposium on*, pages 4–pp. IEEE, 2006.

[27] M. D. Powell and T. Vijaykumar. Pipeline muffling and a priori current ramping: architectural techniques to reduce high-frequency inductive noise. In *Low Power Electronics and Design, 2003. ISLPED'03. Proceedings of the 2003 International Symposium on*, pages 223–228. IEEE, 2003.

[28] S. Prathaban, R. Parthasarathy, and M. C. Falconer. Apparatus, method, and system for predictive power delivery noise reduction, Apr. 1 2014. US Patent 8,689,018.

[29] M. J. Press. *Applied multivariate analysis*. Dover Publications, 2005.

[30] W. R. *Calculus of Variations*. Dover Publications, 1974.

[31] J. Ranieri, A. Vincenzi, A. Chebira, and D. Atienza. Eigenmaps: Algorithms for optimal thermal maps extraction and sensor placement on multicore processors. In *Desgin Automation Conference*, pages 636–641, 2012.

[32] S. Reda, R. Cochran, and A. Nowroz. Improved thermal tracking for processors using hard and soft sensor allocation techniques. *IEEE Transactions on Computers*, 60:841–851, 2011.

[33] V. J. Reddi, M. S. Gupta, G. Holloway, G.-Y. Wei, M. D. Smith, and D. Brooks. Voltage emergency prediction: using signatures to reduce operating margins. In *IEEE 15th International Symposium on High Performance Computer Architecture*, pages 18–29, 2009.

[34] Y. Shi, J. Xiong, H. Chen, and L. He. Runtime resonance noise reduction with current prediction enabled frequency actuator. *Very Large Scale Integration (VLSI) Systems, IEEE Transactions on*, 19(3):508–512, 2011.

[35] Y. Shi, J. Xiong, C. Liu, and L. He. Efficient decoupling capacitance budgeting considering operation and process variations. *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on*, 27(7):1253–1263, 2008.

[36] J. W. Tschanz, S. Narendra, R. Nair, and V. De. Effectiveness of adaptive supply voltage and body bias for reducing impact of parameter variations in low power and high performance microprocessors. *Solid-State Circuits, IEEE Journal of*, 38(5):826–829, 2003.

[37] C. Visweswariah, kaushik Ravindran, K. kalafala, S. G. Walker, S. Narayan, D. K. Beece, J. Piaget, N. Venkateswaran, and J. G. Hemmett. First-order incremental block-based statistical timing analysis. *IEEE Trans. Computer-Aided Design of Integrated Circuits System*, 25:2170–2180, 2006.

[38] C. Visweswariah, kaushik Ravindran, K. kalafala, S. G. Walker, S. Narayan, D. K. Beece, J. Piaget, N. Venkateswaran, and J. G. Hemmett. First-order incremental block-based statistical timing analysis. *IEEE Trans. Computer-Aided Design of Integrated Circuits System*, 25:2170–2180, 2006.

[39] A. Waizman, M. Livshitz, and M. Sotman. Integrated power supply frequency domain impedance meter (ifdim). In *Electrical Performance of Electronic Packaging, 2004. IEEE 13th Topical Meeting on*, pages 217–220. IEEE, 2004.

[40] T. Wang, C. Zhang, J. Xiong, and Y. Shi. Eagle-eye: a near-optimal statistical framework for noise sensor placement. In *Computer-Aided Design (ICCAD), 2013 IEEE/ACM International Conference on*, pages 437–443. IEEE, 2013.

[41] T. Wang, C. Zhang, J. Xiong, and Y. Shi. Eagle-eye: A statistical framework for near-optimal noise sensor placement. In *International Conference on Computer-Aided Design*, 2013.

[42] R. Weekly, S. Chun, A. Haridass, C. O'Reilly, J. Jordan, and F. O'Connell. Optimum design of power distribution system via clock modulation. In *Electrical Performance of Electronic Packaging, 2003*, pages 45–48. IEEE, 2003.

[43] J. Xu, P. Hazucha, M. Huang, P. Aseron, F. Paillet, G. Schrom, J. Tschanz, C. Zhao, V. De, T. Karnik, et al. On-die supply-resonance suppression using band-limited active damping. In *Solid-State Circuits Conference, 2007. ISSCC 2007. Digest of Technical Papers. IEEE International*, pages 286–603. IEEE, 2007.

[44] X. Zhang, J. Lu, Y. Liu, and C.-K. Cheng. Worst-case noise area prediciton of on-chip power distribution network. In *System Level Interconnect Prediction (SLIP), 2014 ACM/IEEE International Workshop on*, pages 1–8. IEEE, 2014.

[45] C. Zhuo, G. Wilke, R. Chakraborty, A. Aydiner, S. Chakravarty, and W.-K. Shih. A silicon-validated methodology for power delivery modeling and simulation. In *Proceedings of the International Conference on Computer-Aided Design*, pages 255–262. ACM, 2012.

VITA

Tao Wang was born in 1989 in Guangxi, China. She received the B.S. degree in Microelectronics (with double major in Statistics) from Peking University, Beijing, China in July, 2011. In August 2015, she received her Ph.D. in Computer Engineering from Missouri University of Science and Technology.

Her primary research interests are statistical modeling and optimization, power integrity and design automation in VLSI design.