Temperature Tracking: An Innovative Run-Time Approach for Hardware Trojan Detection

Domenic Forte
ECE, University of Connecticut
forte@engr.uconn.edu

Chongxi Bao, Ankur Srivastava
ECE, University of Maryland
chongxi.bao@gmail.com, ankurs@umd.edu

***Abstract*—The hardware Trojan threat has motivated development of Trojan detection schemes at all stages of the integrated circuit (IC) lifecycle. While the majority of existing schemes focus on ICs at test-time, there are many unique advantages offered by post-deployment/run-time Trojan detection. However, run-time approaches have been underutilized with prior work highlighting the challenges of implementing them with limited hardware resources. In this paper, we propose innovative low-overhead approaches for run-time Trojan detection which exploit the thermal sensors already available in many modern systems to detect deviations in power/thermal profiles caused by Trojan activation. Simulation results using state-of-the-art tools on publicly available Trojan benchmarks verify that our approaches can detect active Trojans quickly and with few false positives.**

I. Introduction

The emerging trend of outsourcing integrated circuit (IC) design and fabrication has created new vulnerabilities called hardware Trojan attacks that can seriously jeopardize the integrity of an electronic system [1], [2]. Hardware Trojans can change hardware functionality, reduce system reliability, and leak valuable/sensitive information. Since this represents a dangerous threat to all sectors that rely on ICs (military, financial, power grid, etc.), robust methods to detect Trojans are becoming imperative. Unfortunately, Trojan detection happens to be a very challenging problem. Attackers have so many Trojans of varying types and sizes at their disposal [1]. Trojans are also typically triggered by extremely rare events. Finally, measurement noise and fabrication variation can mask Trojan presence [2], [3].

To overcome these challenges, detection approaches have been proposed at three stages of the IC lifecycle: design-time, test-time, and run-time. The majority of existing schemes occur at test-time where the I/O and side channel behavior (delay, power, etc.) of suspect ICs are compared to "golden" models/ICs. Design-time approaches [4], [5] have been utilized (mostly) to support test-time detection. Run-time approaches have been investigated significantly less, but possess several advantages over their counterparts. First, test-time approaches can fail for Trojans with small well-placed triggers (e.g., one gate) that do not get activated at test-time [6]. Run-time approaches on the other hand can be utilized for the *entire lifetime of the IC*. Hence, any Trojan missed during test-time can be found if the Trojan ever gets activated (e.g., [7], [8]). Second, an IC with an inactive Trojan performs essentially the same functions as a Trojan-free IC. Effective run-time detection allows one to still deploy a Trojan-inserted IC and then either disable the IC entirely or bypass the Trojan logic [9], [10] if the Trojan ever gets activated. The main drawback to run-time approaches has been their large overheads [2]. For example, the path delay characterization approach proposed in [7] suffers from considerable area overhead for modern designs with millions of paths [1].

Main Contributions. In this paper, we propose innovative low-overhead approaches for online Trojan detection that take advantage of the relationship between power and temperature. When a Trojan gets activated at run-time, it can have a significant impact on the system's power consumption which will be reflected in the system's thermal profile. Furthermore, while direct monitoring of side channels (current, delay) typically requires significant overheads, thermal sensors and infrastructure are already available in many modern systems for dynamic thermal management[1]. A summary of our main contributions is as follows:

- We propose a novel framework for temperature-based Trojan detection which consists of design-time, test-time, and run-time phases. In the design phase, we statistically characterize an IC's power/thermal dynamics and optimally place thermal sensors. The test-time phase is used to calibrate each IC due to fabrication variation and weed out some of the Trojan-infected ICs. The run-time phase integrates the information from the previous phases with thermal sensor measurements to detect Trojans that are missed during testing and activated at run-time.
- We propose two mechanisms to detect Trojan activation during run-time. The first is a local sensor-based approach that uses information from thermal sensors, statistical information provided by the test phase, and hypothesis testing. The second is a global approach that exploits correlation between sensors and maintains track of the IC's thermal profile using a Kalman filter (KF).
- We test our detection mechanisms on five publicly available Trojan benchmarks (from [12]) and use state-of-the-art Cadence simulation tools to compute power/thermal profiles. In all but one benchmark, the proposed approaches are capable of detecting Trojan activation quickly (less than 210ms) and with very few false-positives.

Organization. In Section II, we give a brief overview of Trojans and Trojan detection. In Section III, we further discuss thermal sensors and the motivation for our temperature-based run-time detection. Our specific problem and its challenges are clearly defined in Section IV. The proposed framework and detection mechanisms are discussed in Section V. Experimental results are discussed in Section VI. We discuss the merits of our approaches in Section VII and conclude in Section VIII.

II. Background

Trojan Anatomy. Trojans consist of two components [1]: (i) The Trojan *trigger* waits for a special event, such as a rare external input pattern or internal logic state, and then activates the Trojan's attack. Before the Trojan is triggered, the IC containing the Trojan functions mainly as intended (excluding

[1]As of this writing, there is only one other paper [11] that explicitly uses temperature to detect Trojans. Their work was developed in parallel with ours and, in contrast to ours, is a test-time Trojan detection approach.

the trigger's activity); (ii) After the Trojan is triggered, the Trojan *payload* is activated and changes the IC's functionality.

In this paper, we refer to ICs with and without Trojans as **Trojan-inserted** and **Trojan-free** respectively. Trojan-inserted ICs whose payloads have been triggered and not triggered are referred to as **Trojan-active** and **Trojan-inactive** respectively.

Trojan Detection Approaches. Hardware Trojans can seriously jeopardize the hardware and software integrity of any electronic system [1], [2]. As a result, there has been a strong initiative in academia, industry, etc. to develop ways to detect hardware Trojans. Detection approaches have been proposed at all three stages of the IC lifecycle: design-time, test-time, and run-time.

1) Test-time: These approaches consist of additional tests that take place after conventional post-manufacturing testing. There are two types. *Logic-based* schemes develop directed test patterns that activate Trojan payloads in order to detect errors in the output. *Side Channel-based* approaches measure physical parameters, such as power consumption and path delay, of suspect ICs, and compares them with expected parameters of a "golden" model or Trojan-free IC. Unlike logic-based, the Trojan payload need not necessarily be activated because the trigger alone may impact IC delay, power, etc.

2) Run-time Monitoring: Run-time approaches also monitor for unexpected changes in logical and side-channel behavior to detect Trojans, but do so after the IC has been deployed (e.g., [7], [8], [10], [13]). Resource overheads have been the main disadvantage of run-time monitoring [2].

3) Design-for-Trust (DFT): These are design-time strategies that aid test-time approaches. Examples include [4] and [5] which use scan flip-flops to increase the probability of Trojan activation and enhance side channel analysis.

Since the above schemes have their own unique advantages and disadvantages, [2] suggests an integrated approach that utilizes all of them to provide more comprehensive coverage of Trojans. Logic and side channel-based detection cover small and large Trojans respectively while run-time monitors act as "last lines of defense" for Trojan-inserted ICs that bypass test-time schemes.

III. Motivation and Discussion

The majority of existing detection schemes occur at test-time, but flaws in test-time approaches can allow Trojan-inserted ICs to end up being deployed.

- *Inactive Trojans.* Inputs to the Trojan trigger are often chosen carefully by the attacker to prevent accidental Trojan triggering during test-time. Since there is only a limited amount of time to perform tests, the Trojan may remain inactive/dormant and, hence, will not cause any changes to the IC's logical behavior.
- *Small Triggers.* Very small triggers (e.g., one gate [6]) can be used in the Trojan attack. Such triggers will have negligible impact on path delay and IC power consumption.
- *Low Impact on Side Channels.* The impact of the Trojan payload on side-channels can easily be masked while the Trojan is inactive. As discussed in [6], the trigger can be used to "power gate" [14] the payload, which effectively minimizes its impact on power consumption and delay.

The above flaws have motivated us to investigate a run-time approach to compliment test-time detection. Simply put, since run-time approaches can be utilized for the *entire lifetime of the IC*, they can overcome many of the shortcomings of test-time approaches. While a Trojan-inserted IC's behavior may be indistinguishable from a Trojan-free IC's while the Trojan is inactive, at some point during run-time the Trojan will have to be activated in order to attack the IC. When a Trojan is activated at run-time, its attack may have *much larger impact* on both logical and side-channel behavior, thereby enabling easier detection.

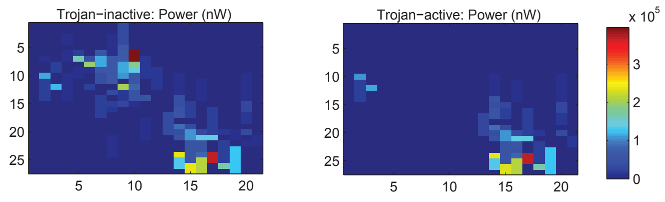

Fig. 1: Avg. power consumption (nW) in a 250μs time window across Trojan-inactive and Trojan-active ICs for the RS232-T900 benchmark.

A. Power Consumption of Active/Inactive Trojan Payload

As a motivating example, we compare power consumption of Trojan-free, Trojan-inactive, and Trojan-active ICs for a publicly available Trojan benchmark: RS232-T900 from trust-HUB [12]. RS232-T900 is a micro-UART core with a Trojan inserted in its transmitter. The Trojan is triggered by a sequence of four transmission messages and its payload prevents any further transmission. We determined power and layout information by simulating, synthesizing, and placing the design with Cadence SimVision, RTL Compiler, and Encounter tools.

The data we collected showed that there was only a 3% difference in total power consumption between Trojan-free and Trojan-inactive ICs. Such a difference is too small to detect at test-time, especially in the presence of measurement noise and process variation. In contrast, we show the power consumption of Trojan-inactive and Trojan-active ICs in Figure 1. Comparing the two cases, one can see that the power differs in the upper left quadrant where the transmitter hardware is located. Transmission is blocked in the Trojan-active IC whereas the Trojan-inactive IC transmits and receives data at all times. Overall, there is a 40% decrease in total power consumption after the Trojan is triggered. Such a change should be significantly easier to observe, thereby enabling better Trojan detection. However, the key challenge is how to detect these changes in power profile inexpensively.

In this paper, we target these Trojans that are well-hidden at test-time (low impact on power), but have significant impact on either the distribution of or the sum total of power consumption once activated at run-time. Our run-time detection schemes detect the changes in power inexpensively by exploiting existing thermal sensors/infrastructure and the relationship between power and temperature.

B. RC Thermal Model

Temperature is a strong function of power consumption. Hence, changes in power caused by an active Trojan should also be reflected in the IC's thermal profile. We illustrate the link between power and temperature using the popular RC thermal model [15]. In the RC model, the IC is divided into grids and the temperature and power consumption of each grid at time t are represented as constants (see Figure 2). A circuit is used to estimate the IC's thermal profile where node voltage and circuit current are analogous to temperature and

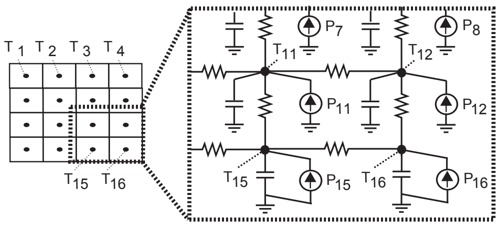

Fig. 2: IC broken into grids and RC thermal model within dotted region

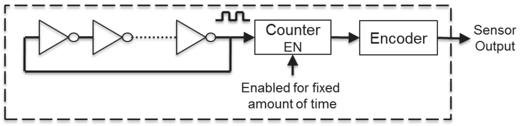

Fig. 3: Basic thermal sensor circuit [22]

power/heat flow in each grid. Voltage ground is analogous to the environment's ambient temperature. Thermal capacitance and thermal resistance between neighboring nodes determine how heat flows between nodes/grids of the IC. Temperature for the entire IC can be determined by solving the following set of ODEs:

$$\sum_{\forall j \in N_i} \frac{1}{R_{ij}} (T_i(t) - T_j(t)) + C_i \frac{dT_i(t)}{dt} - P_i(t) = 0 \quad \forall i \tag{1}$$

where $T_i(t)$ and $P_i(t)$ are the temperature and power dissipated at node i and time t; C_i denotes thermal capacitance at node/grid i; R_{ij} denotes thermal resistance between nodes i and j; and N_i is the set of all neighbors for node/grid i. The above equations are often written in a discrete matrix form [16]:

$$\vec{T}[k] = \mathbf{A}\vec{T}[k-1] + \mathbf{B}\vec{P}[k-1] \tag{2}$$

where $\vec{T}[k]$ and $\vec{P}[k]$ are temperature and power vectors (each element corresponds to one node/grid) at discrete timestep k; $\mathbf{A}$ and $\mathbf{B}$ are coefficient matrices that depend upon the RC circuit and timestep duration. The above formulation is quite flexible and has been used to simulate thermal dynamics as well as perform online temperature tracking [17], [18].

C. Thermal Sensors and Dynamic Thermal Management

Thermal sensors are heavily utilized for dynamic thermal management (DTM) to prevent IC reliability issues [19] and excessive power consumption [20], [15]. For example, the AMD Opteron multicore processor is equipped with 38 thermal sensors [21]. A basic thermal sensor, consisting of a ring oscillator (RO), counter, and encoder, is shown in Figure 3 [22] and functions as follows. The oscillation frequency of the RO is a function of temperature. The counter measures the oscillation frequency by counting the output pulses from the RO over a fixed period of time. The encoder then converts the count to a temperature value. Since it consists of basic digital components, this thermal sensor can easily be added to processor, ASIC, and FPGA designs.

Basic DTM approaches are reactive in nature. If the temperature of a sensor exceeds a predefined threshold (typically around 80^{o}C), DTM will trigger one or more mechanisms to cool the IC at the sensor's location. While reactive approaches can be effective, they also have shortcomings. First, sensors cannot be placed everywhere resulting in low observability and missed hotspots. Second, thermal sensors are susceptible to noise in the voltage supply [14], which will impact the RO's oscillation frequency and cause the sensor's measurement to be noisy. With thermal sensors being used to trigger cooling, the noise may cause certain hotspots to be missed or cause the DTM to overreact.

To overcome the above issues, recent DTM research [17], [18] has tried to combine the sensor measurements with predictive thermal models such as the RC thermal model discussed above. For instance, [18] used steady state Kalman filtering (KF) to track IC temperature with *only five sensors*. The KF essentially combines prior knowledge of power consumption and statistical noise characteristics with run-time sensor measurements in order to optimally estimate the thermal profile *of the entire IC over time.*

D. Advantages of Temperature-based Detection

Few run-time approaches take advantage of side-channel information because of the overheads involved. For example, a path delay characterization approach was proposed in [7] which, while effective, suffers from considerable area overhead for modern designs with millions of paths [1]. The current sensor approach in [8] should also come with significant area and power overheads. Temperature-based Trojan detection is a relatively unexplored avenue which should have lower overheads compared to previous run-time approaches. First, many electronic systems are already equipped with thermal sensors for DTM. Second, prior work has shown that the infrastructure for run-time thermal estimation has low hardware and software overheads as well [18]. In this paper, we exploit the key features of thermal sensors, the RC thermal model, and the KF to detect unexpected changes in IC power/temperature caused by active Trojans.

IV. Problem Definition and Challenges

Our problem is inspired by the example discussed in Section III-A (RS232-T900). We assume that there are three possible states that an electronic system or IC can be in: *Trojan-free*, *Trojan-inactive*, and *Trojan-active*. Each state is defined by a set of statistical characteristics $\mathbf{S}_f$, $\mathbf{S}_i$, $\mathbf{S}_a$ respectively. The Trojan-free and Trojan-inactive characteristics, while not necessarily identical, are close enough such that the Trojan-inserted IC can evade test-time Trojan detection methods (i.e., $\mathbf{S}_f \approx \mathbf{S}_i$). The Trojan-active characteristics $\mathbf{S}_a$ on the other hand differ significantly from the other two. Our goal is a run-time temperature-based approach that can detect changes from $\mathbf{S}_f$ and $\mathbf{S}_i$ to $\mathbf{S}_a$ after the Trojan is activated. Note, we are not concerned with Trojan-inactive ICs since, prior to Trojan activation, they essentially provide the same functionality as Trojan-free ICs. Stated formally, our problem is as follows:

Given two hypotheses of the system's state:

$$\begin{cases} \mathcal{H}_0 & \textit{The state is Trojan-free or Trojan-inactive} \\ \mathcal{H}_1 & \textit{The state is Trojan-active} \end{cases}$$

Use thermal sensor observations to determine if the IC's state (characteristics) correspond to $\mathcal{H}_0$ ($\mathbf{S}_f$, $\mathbf{S}_i$) or $\mathcal{H}_1$ ($\mathbf{S}_a$).

The above problem has various challenges to overcome some of which are specific to temperature tracking and some of which are common to Trojan detection:

- *Golden IC/model.* Most Trojan detection approaches rely on the existence of a "golden model" to distinguish Trojan-free and Trojan-inserted ICs. In our case, we assume that the Trojan-free design is given and from it we can compute $\mathbf{S}_f$ characteristics to function as our golden model.

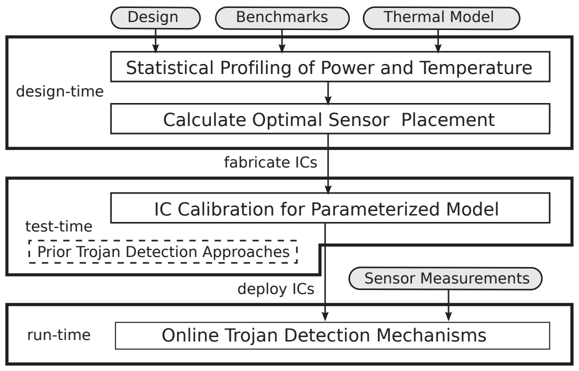
Design
Benchmarks
Thermal Model
design-time
Statistical Profiling of Power and Temperature
Calculate Optimal Sensor Placement
fabricate ICs
test-time
IC Calibration for Parameterized Model
Prior Trojan Detection Approaches
deploy ICs
Sensor Measurements
run-time
Online Trojan Detection Mechanisms

Fig. 4: Phases of the Proposed Approach

- *Autonomous detection.* Since the $\mathbf{S}_f$ characteristics are known and $\mathbf{S}_f \approx \mathbf{S}_i$, we should be able to easily track temperature for Trojan-free and Trojan-inactive designs as in prior work. The challenge is detecting active Trojan ICs because the $\mathbf{S}_a$ characteristics are unknown. We propose two mechanisms for detecting Trojan activation at run-time.
- *Sensor Infrastructure and Noise.* Prior work has shown that sensor placement, number of sensors, sensor noise, etc. have a profound impact on temperature tracking [22]. In this paper, we vary the number of sensors to see the impact on temperature-based Trojan detection. One of our approaches uses the Kalman Filter which explicitly accounts for measurement noise.
- *Fabrication Variation (FV).* FV makes it more challenging to track temperature as well as detect Trojans. For tracking, FV results in larger uncertainty in the estimated thermal profile [18]. For Trojan detection, FV makes it difficult to distinguish between deviations in power/temperature due to manufacturing and Trojan presence [3]. In our framework, calibration is performed for each IC to ensure robustness in the face of FV.

V. Temperature-Based Detection

In this section, we discuss the overall framework and algorithms for our temperature-based Trojan detection. While detection itself occurs at run-time, the approach itself is comprehensive in nature with each phase of the IC supply chain/process playing a critical role. Namely, offline profiling steps at design-time and test-time are used to deal with many of the challenges discussed above. An overview of entire approach with design, test, and run-time phases is shown in Figure 4. The details of each phase are discussed below.

Assumptions. Before we begin, please make note of the following assumptions. As discussed above, side channel-based Trojan detection often relies on a "golden" model or IC. Our approach assumes that we have access to the Trojan-free design or prototype from which we can obtain statistical characteristics ($\mathbf{S}_f$) of switching activity, power consumption, thermal dynamics, etc. For simplicity, we shall also assume that the statistical characteristics are Gaussian. In general, this a valid assumption due to the central limit theorem (CLT) which states that the statistical characteristics of a sufficiently large number of independent random variables will be approximately normally distributed[2].

[2]Note, however, that our detection approaches are very general and can be extended to deal with other statistical characteristics as well.

A. Design Phase

Design Profiling. We profile the Trojan-free designs using the RC thermal model (see Section III-B), benchmarks, and either state-of-the-art simulation tools or prototype ICs. The RC thermal model divides the power consumed by the design into grids and uses a vector $\vec{P}$ to represent power consumed in the grids. The statistical approaches for temperature tracking and Trojan detection used in this paper require probability distribution functions (PDFs) to summarize the design's expected power and/or temperature. Benchmarks that are representative of the design's expected workload along with simulation tools are used to estimate the PDFs. Alternatively, IC prototypes that are verified as Trojan-free can be used. We approximate the PDFs as Gaussian with mean vectors $\vec{\mu}_p$ and $\vec{\mu}_T$ and covariance matrices $\mathbf{Q}_p$ and $\mathbf{Q}_T$ for power and temperature respectively.

Sensor Placement. We leverage the knowledge of statistical characteristics to determine optimal sensor placement for temperature tracking/Trojan detection. Specifically, we adopt the sensor placement approach from [22] which uses the temperature covariance matrix $\mathbf{Q}_T$. Sensors are placed in a greedy fashion to minimize the following cost function

$$cost = \sum_{\forall grids:i} \max\left(0, 1 - \sum_{\forall sensors:j} q_{i,j}\right) \tag{3}$$

where $q_{i,j}$ denotes the element at location i and j of matrix $\mathbf{Q}_T$ (i.e., correlation in temperature between IC grids i and j).

B. Test Phase

When the design phase is complete, we fabricate the ICs. At this point, we assume that some Trojan-inserted ICs are fabricated and inserted into the supply chain.

IC Parameter Calibration. Fabrication variation (FV) results in ICs that have different physical, electrical, and performance parameters from the nominal design. As discussed above, FV makes it more challenging to accurately detect Trojans and track temperature. To ensure robustness, we must have accurate power/thermal statistics ($\vec{\mu}_p$, $\vec{\mu}_T$, $\mathbf{Q}_p$, $\mathbf{Q}_T$) for each IC under test (ICUT). One can accomplish this by applying test vectors to the ICUT, measuring power consumption, and estimating the PDFs after fabrication. For example, gate-level characterization has been applied in prior work [3] to successfully profile IC gate parameters. Temperature-based approaches which utilize infrared cameras [11] and Expectation Maximization (EM) [23] are also applicable.

Test-time Detection. One can argue that our temperature-based approach may not be able to detect all types of Trojans. For example, Trojans that are only active for a few clock cycles may not have a large impact on power and/or temperature. This is why we emphasize an integrated framework as shown in Figure 4. Prior test-time detection schemes are used during this phase to remove Trojan-inserted ICs (that might be missed by our approach) before they are deployed. The proposed run-time approach would then complement test-time approaches by detecting the Trojan-inserted ICs (that are missed by test-time schemes) as they are activated in the field.

C. Run-time Phase

As discussed in Section IV, our main problem is to decide the correct hypothesis (state of the system): $\mathcal{H}_0$ or $\mathcal{H}_1$. In other words, is the IC Trojan-free/Trojan-inactive or Trojan-active? In this section, we propose two mechanisms to solve the problem. The first is a local sensor-based approach that uses an hypothesis testing (HT) framework. The second is a global approach that exploits correlation between sensors and maintains track of the IC's thermal profile with a Kalman filter (KF).

1) Local Sensor Approach (HT): For simplicity, let us suppose we have one sensor measurement at timestep k denoted by $S[k]$ from which we shall decide the state (we'll consider more sensors later). In an hypothesis testing (HT) framework, one assumes that $S[k]$ can only come from one of two PDFs, S_0 or S_1, which correspond to null and alternative hypotheses respectively. In our case, the null and alternative hypotheses correspond to the IC thermal state in Trojan-free/Trojan-inactive ($\mathcal{H}_0$) and Trojan-active ICs ($\mathcal{H}_1$) respectively. We shall choose the correct state as the one with the highest probability of occurrence given $S[k]$ (i.e., $\operatorname{argmax} P(H_x \big| S[k]), x \in \{0,1\}$). By applying Bayesian decision theory, it can be shown the optimal decision is [24]:

Choose $\mathcal{H}_1$ (Trojan-active state) when:

$$\frac{p(S[k]\big|\mathcal{H}_1)}{p(S[k]\big|\mathcal{H}_0)} > \frac{P(\mathcal{H}_0)}{P(\mathcal{H}_1)} \tag{4}$$

Otherwise, choose $\mathcal{H}_0$ (Trojan-inactive state).

where $P(x)$ and $p(x\big|y)$ denote the prior probability of x and probability of x given y respectively.

While the above decision rule is theoretically sound, it is difficult to directly apply to our problem for two reasons.

- One cannot assume that all measurements come from single stationary PDF (i.e., one that is time invariant) since the IC temperature varies with time.
- Even if the PDFs were stationary, we do not have access to the Trojan-active design and therefore cannot accurately estimate $p(S[k]\big|\mathcal{H}_1)$ or $P(\mathcal{H}_1)$.

We get around these issues by making the following simplifying assumptions.

Stable State Temperature. The first issue no longer presents a problem when the IC's temperature has reached a stable state. Put simply, if an IC's power consumption is similar for a long period of time, the IC's temperature will end up converging to a "stable state" [25] where measurements of its thermal state are actually samples from a stationary PDF. In our approach, we assume that we have a verified Trojan-free design/IC (see Section IV) and benchmarks that can characterize the IC's activity. We can then run the benchmarks on the verified Trojan-free design/IC until reaching the stable state. From there, we take measurements and approximate $p(S[k]\big|\mathcal{H}_0)$, as Gaussian with mean μ_0 and variance σ_0^2.

Trojan PDF Estimate: To overcome the second issue, we exploit the fact that S_0 and S_1 must be slightly different and assume a simple Trojan attack model. Specifically, we assume that the mean of $p(S[k]\big|\mathcal{H}_1)$ (referred to as μ_1) differs from $p(S[k]\big|\mathcal{H}_1)$'s known mean ($\mu_0$) by some fixed percentage difference $S_{\%}$ and both possess the same variance ($\sigma_0 = \sigma_1$.) If $S_{\%} < 0$ ($S_{\%} > 0$), Trojan activation causes the IC to lose (gain) some functionality. While this assumption is imperfect, it's simply the best we can do to apply the theory since we have little if any knowledge of the actual Trojan-active design/attack. Moreover, it does allow us to make optimal guarantees for Trojan detection. If the assumed statistics hold, then we can optimally detect the active Trojan. Furthermore, if the $\mathcal{H}_1$ statistics differ from the $\mathcal{H}_0$ statistics by more than $S_{\%}$, we can also probably detect the Trojan.

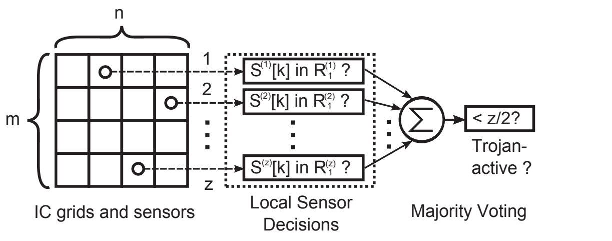

Fig. 5: Local Hypothesis Testing (HT) Approach with z sensors and $m \times n$ grid

Single sensor Decision Rule. With the above assumptions, we now have enough information to apply the optimal decision rule in Eqn. (4). For simplicity, we assume that $P(\mathcal{H}_0) = P(\mathcal{H}_1) = .5$ and express the conditional probabilities with

$$p(S[k]\big|\mathcal{H}_x) = \frac{1}{\sigma_x\sqrt{2\pi}} exp\left(-\frac{(S[k]-\mu_x)^2}{2\sigma_x^2}\right), \quad x \in \{0,1\} \tag{5}$$

With the above, one can easily solve Eqn. (4) w.r.t. thermal measurement $S[k]$ to determine a test statistic [24] (i.e., a hypothesis test specified in terms of temperature)

Choose $\mathcal{H}_1$ (Trojan-active state) when:

$$S[k] \in R_1 \tag{6}$$

Otherwise, choose $\mathcal{H}_0$ (Trojan-inactive state).

In the above equation, R_1 defines a set of temperature values belonging to the Trojan-active state $\mathcal{H}_1$. If the sensor measurement $S[k]$ is from R_1, then the IC's state is most likely $\mathcal{H}_1$. Otherwise, the state is most likely $\mathcal{H}_0$. Note that for the single mode Gaussian PDFs assumed in this paper, Eqn. (6) can be simplified to either $S[k] < R_1$ or $S[k] > R_1$ (depending on the sign of $S_{\%}$). Nevertheless, the theory and associated decision rule are general enough to handle other PDFs as well.

Multi-sensor Decision Rule. For multiple sensors, we can easily extend the above rule by collecting the sensor measurements as a vector $\vec{S}[k]$ and using multivariate Gaussian PDFs. For simplicity, we take a simple ad-hoc approach instead. We evaluate Eqn. (6) for each sensor and come to decision ($\mathcal{H}_0$ or $\mathcal{H}_1$) by majority voting.

Overheads. A high-level overview of the local HT approach and its overheads is shown in Figure 5. HT's offline overhead includes computing the optimal test-statistics (R_1) for each sensor. At run-time, a simple circuit determines if the sensor measurement is in the sensor's corresponding R_1 and outputs a 1-bit vote. Note that for the single mode Gaussian PDFs assumed in this paper, the circuit one needs to implement is a simple comparator. Majority voting is used to combine the decisions of z sensors and obtain a final decision. All these operations are simple and the overall complexity *depends only on the number of sensors z.*

2) Global (Kalman Filter-based) Approach: This is a global sensor-based approach that exploits correlation between sensors and uses a Kalman Filter (KF) to dynamically track the system's thermal profile at run-time. An autocorrelation based metric then decides between hypotheses $\mathcal{H}_0$ and $\mathcal{H}_1$ (Trojan-free/Trojan-inactive and Trojan-active).

Temperature Tracking Via Kalman Filter. We track IC temperature at run-time using the standard Kalman filtering (KF) approach developed in prior work [17], [18]. For simplicity, we only consider dynamic power, but leakage power can be handled as well (see [26]). The KF relies on a state-space equation to model the random dynamics of the state being estimated and on a measurement equation to relate measurements with the state being estimated. The state-space equation for temperature tracking is the discrete form RC thermal model equation discussed in Section III-B (copied below for convenience)

$$\vec{T}[k] = \mathbf{A}\vec{T}[k-1] + \mathbf{B}\vec{P}[k-1] \tag{7}$$

The above equation assumes that the current thermal state $\vec{T}[k]$ depends on the previous thermal state $\vec{T}[k-1]$ (Markovian assumption) and also local power dissipation $\vec{P}[k-1]$. Due to variations in the voltage supply noise, system workload, etc., the power $\vec{P}$ is random at each timestep and $\vec{T}[k]$ cannot be precisely computed with the state-space model alone. To improve the estimate, the KF uses measurements collected by thermal sensors and the following measurement model

$$\vec{S}[k] = \mathbf{H}\vec{T}[k] + \vec{v}[k] \tag{8}$$

where $\vec{S}[k]$ is a vector of sensor measurements at timestep k; $\mathbf{H}$ is a transformation matrix based on the sensor placement; and $\vec{v}[k]$ is a Gaussian random vector with zero mean and known covariance $\mathbf{R}$ [17] representing measurement noise.

The KF estimates the thermal state of a chip as follows. $\vec{P}[k]$ is modeled as a Gaussian random vector[3] with known mean $\vec{\mu}_p$ and covariance $\mathbf{Q}_p$ (which we determine in the design/test phases). KF estimation is then performed recursively with *predict* and *update* steps. In the *predict* step, the KF uses $\vec{\mu}_p$ and the previous temperature estimate to predict the IC's new thermal state. In the *update* step, the KF corrects this estimate based on new sensor measurements. The following equations are used at each timestep

predict:

$$\vec{T}[k|k-1] = \mathbf{A}\vec{T}[k|k-1] + \mathbf{B}\vec{\mu}_p \tag{9}$$

$$\mathbf{C}[k|k-1] = \mathbf{A}\mathbf{C}[k|k-1]\mathbf{A}^T + \mathbf{B}\mathbf{Q}_p\mathbf{B}^T \tag{10}$$

update:

$$\vec{e}[k] = \vec{S}[k] - \mathbf{H}\vec{T}[k|k-1] \tag{11}$$

$$\vec{T}[k|k] = \vec{T}[k|k-1] + \mathbf{K}[k]\vec{e}[k] \tag{12}$$

$$\mathbf{K}[k] = \mathbf{C}[k|k-1]\mathbf{H}^T(\mathbf{R} + \mathbf{H}\mathbf{C}[k|k-1]\mathbf{H}^T)^{-1} \tag{13}$$

$$\mathbf{C}[k|k] = (\mathbf{I} - \mathbf{K}[k]\mathbf{H})\mathbf{C}[k|k-1] \tag{14}$$

$\vec{T}[k|k]$ and $\vec{T}[k|k-1]$ are estimates of the temperature at time k computed with and without sensor information respectively; $\mathbf{C}[k|k-1]$ and $\mathbf{C}[k|k]$ are the error covariance matrices associated with $\vec{T}[k|k-1]$ and $\vec{T}[k|k]$; $\vec{e}[k]$ is the KF *residual* which reflects the discrepancy between the predicted and actual measurements; $\mathbf{I}$ is the identity matrix; $\mathbf{K}[k]$ represents the Kalman gain at the kth step and is chosen to minimize the error in $\vec{T}[k|k]$.

Steady State Kalman Filter. When the statistical characteristics of $\vec{P}$ and measurement noise $\mathbf{R}$ are fixed (or do not change for a relatively long time), the KF stabilizes which means $\mathbf{C}[k|k-1], \mathbf{C}[k|k]$, and $\mathbf{K}[k]$ converge to static values. This is referred to as the KF *steady state*. During the steady state, even though the temperature may change with time, the error associated with the estimates remains the same. The steady state allows one to create low overhead implementations of the KF [18] by replacing $\mathbf{C}[k|k-1], \mathbf{C}[k|k]$, and $\mathbf{K}[k]$ in Eqns. (9) to (14) by constants.

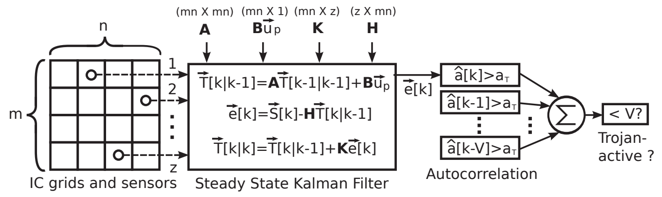

Fig. 6: Global Kalman Filter (KF) approach with z sensors and $m \times n$ grid

Autocorrelation-based Detection Rule. While the KF can be used to accurately track temperature, we also need a rule to decide on the correct state ($\mathcal{H}_0$ or $\mathcal{H}_1$). Our decision rule is based on the KF residual and uses the autocorrelation function of the residual process. In the KF, residual $\vec{e}[k]$ represents the discrepancy between the predicated temperature and thermal sensor measurements. If $\vec{e}[k]$ is small (large), the two agree (disagree) on the thermal state. Assuming the state-space model/parameters and the sensor noise covariance are reasonably accurate, the autocorrelation of the residual should be close to zero on average [27]. When a Trojan gets activated, the state-space model (*which does not account for the power of an active Trojan*) becomes less accurate and should cause the autocorrelation to diverge from zero.

We exploit this behavior and use the following method to detect a Trojan at timestep x. We record the residual in the N previous timesteps of the KF ($\vec{e}[x-N], \vec{e}[x-N+1], \ldots, \vec{e}[x]$) and then compute the following cost function

$$\hat{a}[x] = \tfrac{1}{N}\textstyle\sum_{i=x-N+1}^{x}\left(\vec{e}[i]\cdot\vec{e}[i-1]^T\right) \tag{15}$$

This cost is the average autocorrelation of the residual process in the last N timesteps. To decide if a Trojan is activated, we define thresholds a_T and V. If $(|\hat{a}[x]| > a_T)$ for more than V consecutive timesteps, we assume a Trojan has been activated (i.e., state $\mathcal{H}_1$). a_T and V are parameters that tune the aggressiveness of the decision rule.

Overheads. A high-level overview of the global approach and its overheads is shown in Figure 6. The main offline overheads are estimating the power statistics ($\vec{\mu}_p$ and $\mathbf{Q}_p$) and computing the steady state Kalman gain matrix $\mathbf{K}$. During run-time, the KF performs some matrix-vector multiplications and vector additions/subtractions (Eqns. (9), (11), (12)) at each timestep k. The size of the matrices/vectors and complexity of these operations *depend on the number of grids in the RC thermal model (mn) and the number of sensors (z)*. Running averages of the autocorrelation $\hat{a}$ for the last V timesteps must kept and the final state is chosen based on $V+1$ thresholding operations.

3) Qualitative Comparison: The salient differences between the above two detection mechanisms are as follows:

- *Stable State Assumption:* The local hypothesis testing (HT) approach requires the system under test to be in a stable thermal state so that the sensor measurements can be compared with stationary PDFs. The global Kalman Filter (KF) approach works with the system in any thermal state since it tracks the system's thermal profile at all timesteps.
- *Sensor Correlation:* The local HT approach compares each sensor measurement with its corresponding stable state PDF in an independent fashion. Correlation between the

[3]Note that while Gaussian distributions are assumed for power/temperature by the KF in this paper, prior work [16], [23] has shown that the KF framework can be extended to handle a mixture of Gaussians (MOGs) for more general PDFs. Such approaches can easily account for multiple power profiles, modes of IC operation, etc. For simplicity, we assume a single Gaussian PDF in this paper, but shall evaluate MOGs for Trojan detection in future work.

sensors is not exploited and the final decision is made based on a majority vote. The global KF approach exploits the correlation between sensors to accurately track temperature and detect Trojans.

- *Run-time Overheads:* The KF approach clearly has larger run-time overheads than the HT approach (see Figures 5 and 6). While the HT approach primarily works with scalar values and computes 1-bit decisions, the KF approach requires matrix-vector storage and computations which are more expensive ($O(mn)$).

VI. Experiments and Discussion

A. Setup

Benchmarks. We tested our Trojan detection schemes on five publicly available Trojan benchmarks (from trust-HUB [12]):

1) *RS232-T900* was discussed in Section III-A.
2) *s38417-T300* contains a Trojan trigger with activation probability of 1.7e-44. Once activated, the payload leaks the value of a specific net through a 29 stage ring-oscillator.
3) *BasicRSA-T200* is an RSA encoder with a Trojan triggered by a specific plaintext input. The payload permanently disables encoding of the plaintext.
4) *MC8051-T300* is an implementation of the 8051 microprocessor with a Trojan. The Trojan is triggered when a specific string is sent through the UART and the payload blocks new messages from the UART.
5) *MC8051-T600* is another implementation of the 8051 microprocessor with a Trojan. The Trojan is activated by an external interrupt and disables 8051 instructions containing jumps.

We determined power and layout information in the above benchmarks by simulating, synthesizing, and placing each design with Cadence SimVision, RTL Compiler, and Encounter tools for two different testbench instances: one which activates the Trojan (i.e., Trojan-active) and one which does not (i.e., Trojan-inactive). The difference in power consumption between the two is shown in Table I for all the benchmarks. In all cases but one (MC8051-T300) there is a % difference larger than 25%.

Temperature-based Trojan Detection. We divided the IC into 20 by 16 grids (320 distinct regions). In the design phase, we computed power and temperature statistics using 250ms of data generated by Cadence (from the Trojan-free designs) and the RC thermal model [15]. With the resulting statistics, we placed sensors as discussed in Section V-A. The number of sensors we tested were 4, 16, and 32. For simplicity, we ignored fabrication variation and therefore did not implement the test phase. For the run-time phase, we computed "real" dynamic thermal profiles using the RC thermal model. A steady state Kalman filter (KF) implementation was used to estimate the thermal profile for Trojan-active and Trojan-inactive cases. Sensor measurements were made by overlaying noise onto the "real" thermal profile. We assumed sensor noise variance of 0.1 which seems like a worst-case for state-of-the-art thermal sensors [28]. For KF-based Trojan detection, we stored $N = 50$ residuals and chose autocorrelation thresholds $V = 10$ and $a_T = .18, .34, .40$ for 4, 16, and 32 sensors respectively (based on data from the Trojan-free designs). For the hypothesis testing (HT) approach, we used mean difference $S_{\%} = \pm 2.5\%$ to estimate the Trojan-active stable state PDFs ($\mathcal{H}_1$ case). Except where specified, one can assume

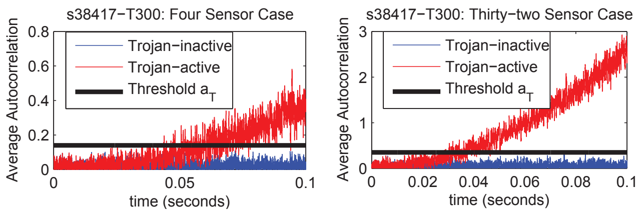

Fig. 7: Average autocorrelation ($\hat{a}$) over time with 4 and 32 sensors for s38417-T300

the experiments were conducted while the ICs were in stable thermal states.

B. Results

We conducted 100 trials with random sensor noise on both the Trojan-inactive and Trojan-active ICs. We recorded the following data: average true positive rate t_+, average false positive rate f_+, and average time t_{avg} to obtain a true positive. The results are shown for all 5 benchmarks and both detection mechanisms in Table II. "HT" and "KF" denote the local hypothesis testing and global Kalman Filter based approaches respectively.

Local Hypothesis Testing (HT). HT was able to detect the active Trojans (true positives) in all the benchmarks with 100% accuracy except for MC8051-T300. MC8051-T300 had the smallest difference in power consumption between Trojan-active and Trojan-inactive cases (see Table I) and did not yield enough deviation in the thermal profile for detection. False positives on the inactive Trojans were only an issue in the 4 sensor case and went to 0% with additional sensors. Increasing the number of sensors also resulted in slower Trojan detection. Put simply, it took a longer time for the majority of sensors to agree on a true positive when there were more sensors. On average, 4 sensors could detect Trojans 70% faster than 32 (ignoring MC8051-T300), but with higher false positive rate.

Global Kalman Filtering (KF). The KF was also very successful with true positives in every benchmark but one. Once again, the deviation in power/temperature was too small to detect for MC8051-T300. The KF had a false positive rate of zero in all instances but MC8051-T300. In contrast to HT, increasing the number of sensors from 4 to 32 improved detection time by 38% on average (ignoring MC8051-T300). Basically, the more measurements the better the resolution of thermal profile and autocorrelation $\hat{a}$. To illustrate, Figure 7 shows autocorrelation $\hat{a}$ of Trojan-inactive (blue) and Trojan-active (red) ICs with 4 and 32 sensors for s38417-T300. The Trojan-inactive autocorrelation stays below the threshold a_T (black line) while the Trojan-active autocorrelation diverges from zero and exceeds the threshold. The Trojan-active case crosses the threshold more quickly in the 32 sensor case.

Note that the KF results in Table II correspond to the case where the ICs were in stable thermal states, but we also ran trials at room temperature which yielded similar results.

Comparing Local HT and Global KF. While both approaches worked well, the KF achieved better results. The KF found active Trojans 60% faster on average than HT and was effective even with only 4 sensors. Also, while the HT approach could only operate with the ICs in a stable thermal state, the KF worked in all scenarios. The advantage of HT is its lower overheads.

TABLE I: % difference in total power consumption between Trojan-inactive and Trojan-active in 250ms experiment

Benchmark	% difference
RS232-T900	-39.97%
s38417-T300	54.33%
BasicRSA-T200	-28.40%
MC8051-T300	-1.5%
MC8051-T600	-72.16%

TABLE II: Average true positive rate t_+, false positive rate f_+, and detection time t_{avg} (seconds) for 100 trials. Note detection time is given in seconds and only includes true positives. '-' indicates no true positives.

		RS232-T900			s38417-T300			BasicRSA-T200			MC8051-T300			MC8051-T600		
	# sensors	t_+	f_+	t_{avg}	t_+	f_+	t_{avg}	t_+	f_+	t_{avg}	t_+	f_+	t_{avg}	t_+	f_+	t_{avg}
HT	4	100%	79%	4.9E-2	100%	45%	1.9E-3	100%	66%	1.7E-2	0%	0%	-	100%	66%	5.2E-2
	16	100%	0%	1.7E-1	100%	0%	4.3E-3	100%	0%	3.9E-2	0%	0%	-	100%	0%	1.6E-1
	32	100%	0%	2.1E-1	100%	0%	5.1E-3	100%	0%	4.8E-2	0%	0%	-	100%	0%	2.0E-1
KF	4	100%	0%	1.0E-3	100%	0%	3.6E-3	100%	0%	8.3E-3	1%	1%	1.2E-2	100%	0%	3.7E-2
	16	100%	0%	8.3E-4	100%	0%	2.7E-3	100%	0%	6.3E-3	3%	3%	1.1E-2	100%	0%	3.2E-2
	32	100%	0%	5.9E-4	100%	0%	2.2E-3	100%	0%	5.2E-3	7%	7%	1.0E-2	100%	0%	2.4E-2

VII. Summary of Merits

The merits of our framework and approaches are summarized as follows:

- *Exploit Thermal Sensors For Low Sensing Overhead:* Our approaches exploit the fact that Trojan activation can cause significant changes in power consumption, which will be reflected in the IC's thermal profile. Temperature-based Trojan detection has lower sensing overhead compared to prior run-time approaches because many electronic systems are already equipped with thermal sensors for DTM.
- *Strong Theoretical Foundations:* Existing methods for Trojan detection tend to be ad-hoc in nature. They fail to make optimal decisions regarding Trojan presence and do not directly account for measurement/process noise. In contrast, the detection schemes that we propose are more rigorous and rely on fundamental theories from signal estimation and detection theory (namely, Kalman Filter theory, Bayesian decision theory, and autocorrelation metrics).
- *Integrated Nature:* We use an integrated framework where test-time and run-time approaches are used to complement each other. Our temperature-based approach can detect Trojans that have large impact on delay, power, temperature, etc. when triggered at run-time and are inactive at test-time. While our approach can deal with Trojans that are well-hidden, it may miss out on Trojans that are less aggressive (i.e., have low impact on temperature). Hence, we rely on prior test-time approaches to weed out such Trojan-infected designs before they are deployed.
- *Highly Generalizable:* Our approaches are highly generalizable in various respects. First, the thermal sensing circuits (e.g., ring oscillators) can be added to processor, ASIC, and FPGA platforms. Second, the KF approach itself can be extended to utilize multiple and heterogeneous sensing modalities for better IC temperature or state tracking. Finally, our approaches should also be able to handle software Trojan attacks, which may have an even greater impact on power/temperature profiles.

VIII. Conclusion

In this paper, we have proposed novel temperature-based approaches for online Trojan detection. The experimental results on real Trojan benchmarks showed that our proposed schemes were effective in all instances but one. Failure only occurred when the deviation in power/temperature was too small to detect. While we find these results promising, we shall investigate enhancements to the proposed approaches in future work (e.g., heterogeneous sensing).

Acknowledgment

This material is based upon work supported by the National Science Foundation under Grant No. 1223233.

References

[1] M. Tehranipoor and F. Koushanfar, "A survey of hardware trojan taxonomy and detection," *IEEE Des. Test*, 2010.

[2] S. Bhunia, M. Abramovici, D. Agrawal, P. Bradley, M. Hsiao, J. Plusquellic, and M. Tehranipoor, "Protection against hardware trojan attacks: Towards a comprehensive solution," *IEEE Des. Test*, 2012.

[3] F. Koushanfar and A. Mirhoseini, "A unified framework for multimodal submodular integrated circuits trojan detection," *IEEE TIFS*, 2011.

[4] H. Salmani, M. Tehranipoor, and J. Plusquellic, "A novel technique for improving hardware trojan detection and reducing trojan activation time," *IEEE TVLSI*, 2012.

[5] ——, "A layout-aware approach for improving localized switching to detect hardware trojans in integrated circuits," in *Proc. WIFS*, 2010.

[6] S. Wei, K. Li, F. Koushanfar, and M. Potkonjak, "Hardware trojan horse benchmark via optimal creation and placement of malicious circuitry," in *Proc. DAC*, 2012.

[7] J. Li and J. Lach, "At-speed delay characterization for ic authentication and trojan horse detection," in *Proc. HOST*, 2008.

[8] S. Narasimhan, W. Yueh, X. Wang, S. Mukhopadhyay, and S. Bhunia, "Improving ic security against trojan attacks through integration of security monitors," *IEEE Des. Test*, 2012.

[9] M. Abramovici and P. Bradley, "Integrated circuit security: new threats and solutions," in *Proc. Annual CSIIR Workshop*, 2009.

[10] M. Hicks, M. Finnicum, S. King, M. Martin, and J. Smith, "Overcoming an untrusted computing base: Detecting and removing malicious hardware automatically," in *IEEE Symp. Security and Privacy*, 2010.

[11] K. Hu, A. N. Nowroz, S. Reda, and F. Koushanfar, "High-sensitivity hardware trojan detection using multimodal characterization," in *Proc. DATE*, 2013.

[12] trust HUB.org, http://trust-hub.org/resources/benchmarks.

[13] M. Bilzor, T. Huffmire, C. Irvine, and T. Levin, "Evaluating security requirements in a general-purpose processor by combining assertion checkers with code coverage," in *Proc. HOST*, 2012.

[14] N. H. Weste and D. M. Harris, *Principles of CMOS VLSI design: a systems perspective, Fourth Edition*. Addison-Wesley Publishing Company, 2011.

[15] K. Skadron, M. Stan, K. Sankaranarayanan, W. Huang, S. Velusamy, and D. Tarjan, "Temperature-aware microarchitecture: Modeling and implementation," *ACM TACO*, 2004.

[16] Y. Zhang and A. Srivastava, "Adaptive and autonomous thermal tracking for high performance computing systems," in *Proc. DAC*, 2010.

[17] S. Sharifi, C. Liu, and T. Rosing, "Accurate temperature estimation for efficient thermal managemen," *Proc. ISQED*, 2008.

[18] Y. Zhang, A. Srivastava, and M. Zahran, "On-chip sensor-driven efficient thermal profile estimation algorithms," *ACM TODAES*, 2010.

[19] J. Srinivasan, S. V. Adve, P. Bose, and J. A. Rivers, "The impact of technology scaling on lifetime reliability," in *Proc. DNS*, 2004.

[20] R. Rao, S. Vrudhula, and D. N. Rakhmatov, "Battery modeling for energy aware system design," *Computer*, 2003.

[21] Y. Zhang and A. Srivastava, "Accurate temperature estimation using noisy thermal sensors," in *Proc. DAC*, 2009.

[22] Y. Zhang, B. Shi, and A. Srivastava, "A statistical framework for designing on-chip thermal sensing infrastructure in nano-scale systems," in *Proc. ISPD*, 2010.

[23] Y. Zhang and A. Srivastava, "Statistical characterization of chip power behavior at post-fabrication stage," in *Proc. IGCC*, 2011.

[24] S. Kay, *Fundamentals of Statistical Signal Processing: Detection theory*. Prentice-Hall PTR, 1998.

[25] D. Forte and A. Srivastava, "Energy and thermal-aware video coding via encoder/decoder workload balancing," in *Proc. ISLPED*, 2010.

[26] Y. Zhang and A. Srivastava, "Leakage-aware kalman filter for accurate temperature tracking," in *Proc. IGCC*, 2011.

[27] J. L. Crassidis and J. L. Junkins, *Optimal Estimation of Dynamic Systems*. CRC Press, 2004.

[28] F. Sebastiano, L. Breems, K. Makinwa, S. Drago, D. Leenaerts, and B. Nauta, "A 1.2 v 10μw npn-based temperature sensor in 65nm cmos with an inaccuracy of $\pm 0.2^o$c from -70^{o}c to 125^{o}c," in *Proc. ISSCC*, 2010.