Quantifying Process Variations and Its Impacts on Smartphones

Guru Prasad Srinivasa*, Scott Haseley†, Mark Hempstead‡, and Geoffrey Challen†
*University at Buffalo, †University of Illinois at Urbana-Champaign, ‡Tufts University

***Abstract*—Process variation can cause the performance and energy consumption of smartphones of the same model to vary significantly. While process variation has been studied in detail, the effects on smartphone performance have not been quantified and evaluated. In this work we study the performance and energy differences of 5 recent SoC generations caused by underlying process variation.**

We make two important contributions. First, we present a methodology to construct a temperature-stabilized environment to perform repeatable power and performance measurements. Studying power-performance characteristics of smartphones is difficult. Running a benchmark back-to-back often produces significantly different results due to heat. Temperature, both device and ambient, play a significant role in determining performance and energy. Our methodology allows us to control for various factors and isolate the effects of the underlying process variation. We then apply our methodology to investigate performance and energy characteristics of several recent generations of smartphone CPUs that result from process variation. Our results show that devices of the same model may exhibit differences of 10% and 12% difference in performance and energy over a fixed-duration workload.

I. Introduction

Process variation has been extensively studied over the past few decades and its sources are well-understood [1], [2], [3], [4]. It is even being exploited on large multicore chips [5] and dark silicon chip multi-processors [6] to compute optimal mappings for threads and cores. But on the ubiquitous smartphones that are in production and in users' hands today, the effects and consequences of process variation are less studied and less known.

No two chips are produced equal. This is an inescapable fact of the chip manufacturing process, and increasing chip complexity and reducing transistor sizes have exacerbated this inequality. Variations in the underlying transistors can cause devices of the *same* make and model to exhibit different thermal characteristics. This in turn can lead to some devices to heat up more quickly which forces them to slow down. As a result, your smartphone could be up to 20% worse in energy *and* performance than other devices of the same make and model [7]. Unlike desktop processors, where significant process variations result in frequency and pricing differences, the mobile market appears to paper over them. Consequently, chips of varying degrees of quality find their way into devices that are identical in appearance and price.

By ensuring that all their CPUs operate at the same frequency—a process known as voltage binning [8]—manufacturers create chips that are indistinguishable to an unassuming consumer. In this process, all CPUs are configured to have the same operating frequency while their individual supply voltages are tweaked as necessary to ensure stable operation. From the consumer's perspective, their phone appears to be running just as fast as any other phone of the same model. Behind the scenes, however, the phone may be consuming more energy to do so, converting electrical energy to thermal energy and thus heating up in the users hand.

For decades, system builders have effectively hidden these effects through the use of active cooling, careful case design, and controlled thermal environments. Unfortunately, smartphones frustrate all of the strategies established to gain thermal control over hot CPUs. Unlike stationary devices and even larger mobile devices, smartphones get used in uncontrolled thermal environments—everywhere from hot cars to cold winter nights. And smartphones are too small to incorporate the active cooling components commonly found on servers, desktops, and laptops, such as fans or large heat sinks. At their top frequencies, the heat generated by smartphone CPUs can reach their thermal limits within seconds of beginning a compute-intensive task. Once these limits are reached, throttling strategies such as disabling cores or reducing CPU frequencies must be used to cool the device, which in turn degrades performance.

Prior to the advent of ARM's big.LITTLE, chips would often expose their binning information in the form of a number. Smartphone enthusiasts would use these bin numbers to determine the voltages that the chip operated at, and in some cases end up drawing the wrong conclusions [9]. Srinivasa et al [7] described how, contrary to popular belief, devices that operated at the highest voltages were often the most performant. Currently, to the best of our knowledge, chips no longer expose such binning information. While variations in voltage-frequency tables continue to exist, they are often hidden from the end-user.

To understand the characteristics of their smartphones, users today resort to running benchmarks and comparing results. In the best case scenario, comparisons can be made against devices of the same model as that of the user's. In the worst case, users are left with a score and the scores of the top 50 device models overall [10]—a list in which their model may not even figure. Even in the best case scenario, the results are skewed in favor of lower ambient temperatures. The score of a good CPU would be no match to the score of a bad CPU if the bad CPU ran the benchmark at a significantly lower ambient temperature. Guo et al [11] discusses how putting a

smartphone in a refrigerator could improve the overall score of Antutu [12], a popular benchmark, by more than 60%. Furthermore, running the same benchmark back-to-back would yield significantly different scores as the second run begins with a warm device—a consequence of the first run.

In this work, we measure and evaluate the performance and energy differences caused by process variation on several recent generations of smartphone SoCs. While smartphone manufacturers and power users may have moved on to the latest SoC, many others including researchers continue to use SoCs that were released a few years ago. The Nexus 5, first released in 2013, continues to be in the top 10 active devices that use the open-source LineageOS [13], a fork of the popular Cyanogenmod operating system.

We make the following contributions as part of this work:

- An experimental methodology and setup that is capable of identifying and measuring process variation on smartphone at the system level (§ III).
- Experimental analysis and results of 5 out of the possible 8 generations of Qualcomm SoCs released since 2013 (§ IV) In particular our results show that process variation led to energy-performance variations as high as 20% in older SoCs. While less profound, it continues to be significant in recent smartphone SoCs with observed performance and energy variations of 5% and 10% respectively.

II. Background & Motivation

Our interest in studying process variations arose from our inability to reproduce performance results that we had earlier observed while running a CPU intensive benchmark. By swapping the SoC while keeping the workload, device casing, and battery constant, we confirmed that the SoC was the source of variation. Further experimentation revealed that the variations were caused due to intrinsic properties of the SoC.

The process of segregating CPUs based on its manufacturing quality and electrical characteristics is known as *CPU binning*. Note that because multiple cores that are part of a single CPU are drawn from the same patch of silicon, differences are between entire CPUs and not between cores. The two major binning techniques used by manufacturers are speed binning and voltage binning. When chips are manufactured, they are first tested to identify their stable operating frequencies. If a chip does not meet the necessary timing constraints or fails to operate at the expected frequency, the operating frequency is lowered until it passes the tests. The chips are then sorted into bins and labeled according to their speed. This process is called speed binning. They are then sold at price points proportional to their speed bin [8].

Speed binning labels chips according to their speed. Voltage binning keeps the frequency fixed across all chips and adjusts the voltage across bins. Voltage binning is based on the fact that both speed and leakage power of a transistor are a function of the supply voltage. Slow transistors—ones with larger gate lengths—leak less, while fast transistors—ones with shorter gate lengths—leak more. Manufacturers thus divide the chips into voltage bins where slower chips are binned at higher voltage so as to support the required operating frequency, while faster chips are binned at lower voltage in order to reduce their already high energy consumption. We believe that this is done in order to try and provide consistent performance (in terms of speed) across all devices using the same SoC.

Table I lists voltages used for multiple frequencies across bins on a Nexus 5 device. Bin-0 has the slowest transistors while bin-6 transistors leak the most. Therefore, bin-6 operates at lowest voltage while bin-0 voltage is increased to enable equal performance as bin-6. Manufacturers thus use this technique to attempt to enable consistent performance across all bins. Note that the process controls for speed, so both the bin-0 and bin-6 CPUs provide the same set of operating frequencies.

Voltage	Frequency (MHz)				
(mV)	300	729	960	1574	2265
Bin-0	800	835	865	965	1100
Bin-1	800	820	850	945	1075
Bin-2	775	805	835	925	1050
Bin-3	775	790	820	910	1025
Bin-4	775	780	810	895	1000
Bin-5	750	770	800	880	975
Bin-6	750	760	790	870	950

TABLE I: **Voltage vs. Frequency across bins.** Voltages for various frequency levels across bins for Nexus 5 as listed in kernel sources.

Different transistor properties combined with varying operating voltages leads to differences in the thermal characteristics between various CPU bins. Despite the manufacturer's efforts, these thermal characteristics in turn result in variations in *both* energy consumption and performance. Figure 1 describes the energy characteristics of the different CPU bins on the Nexus 5. It plots the energy consumption of various Nexus 5 bins while performing a fixed CPU intensive workload. From the figure, we see that bin-4 consumes about 20% more energy than bin-0 while also taking ≈20% more time to do the same amount of work due to thermal throttling.

Ambient temperature also plays a crucial role in determining the amount of energy consumed to do a certain amount of work. The leakage current of transistors is proportional to temperature [14]. Transistors that leak more also generate heat at faster rate compared to those with lower leakage. To make matters worse, in cases where the cooling rate is not increased, the higher heat dissipation increases the temperature of the device which in turn creates a feedback loop that increases leakage current. Figure 2 describes this trend for two different devices. Both devices consume up to 30% additional energy to do the same work at higher ambient temperatures.

Being aware of the differences between seemingly-identical devices of the same make and model is important, but being able to identify them is paramount. Detecting these differences will benefit researchers who run experiments on a small set of devices and extrapolate their results to larger sets, and can help consumers understand the range of quality for a particular device model.

III. Design & Methodology

Existing benchmarks are insufficient to measure underlying transistor differences as they don't consider temperatures—

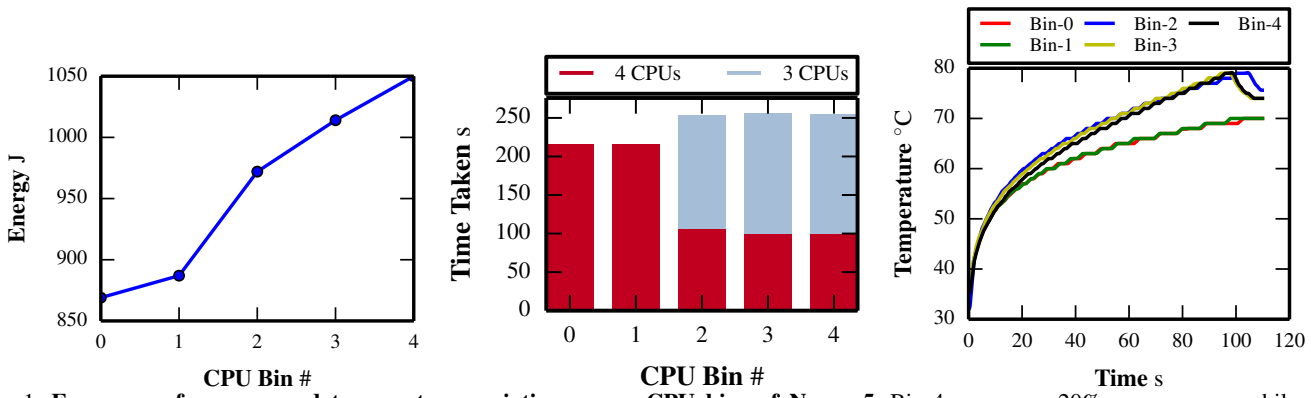


Fig. 1: **Energy, performance and temperature variation across CPU bins of Nexus 5**. Bin-4 consumes 20% more energy while also taking 18% longer due to thermal throttling. Once thermal limits of 80°C are reached, one CPU core is shut down.

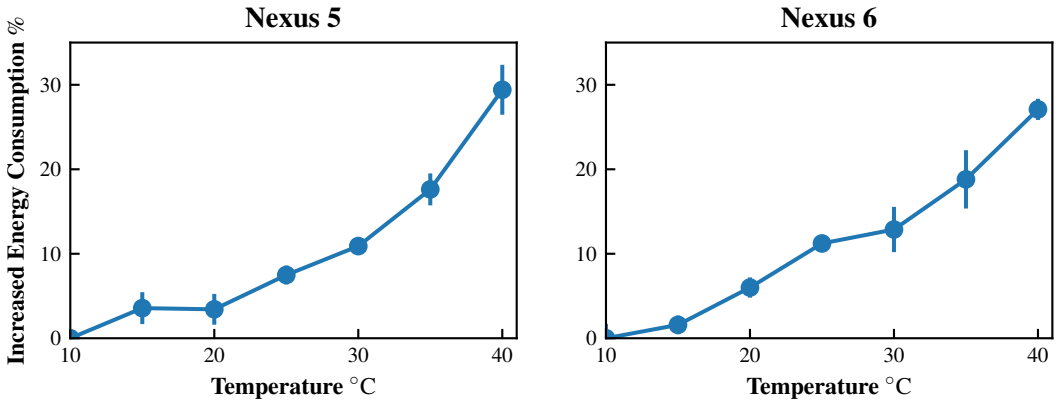


Fig. 2: **Energy scaling on two different devices at max frequency**. Differences in ambient temperature can cause an increase of 25% or more energy consumption to do the same work. This effect is observed across devices.

neither device nor ambient. We designed our benchmarking technique, ACCUBENCH, to reliably quantify the energy-performance characteristics of smartphone CPUs and attempt to expose the underlying transistor differences. We expose transistor variations by running a CPU intensive workload and comparing its results with those from other devices of the same model. The idea here is that a CPU with bad transistors would generate more heat and thereby yield lower performance thus scoring less in our CPU intensive workload.

The ACCUBENCH technique can be broken down as follows:

- Warm up the CPU for fixed time
- Perform cooldown until CPU reports target temperature
- Run workload for fixed time

A problem with existing benchmarks is that they produce very different results on the same CPU depending on whether the CPU was previously idle or under use. The warmup phase mitigates this by synthetically generating heat and warming up the CPU. Thus, CPUs that were idle become warm while CPUs that were previously warm remain so. The cooldown phase ensures that the workload phases of all experimental iterations across devices are run under similar thermal states. Finally, the main CPU-intensive workload is executed.

The entire technique is packaged into an app that could be invoked via an Android intent. At its core, our app uses a WebView and all of the core functionality was written in JavaScript. This JavaScript code uses APIs exposed by the app to perform restricted operations such as reading the CPU temperature, acquiring wakelocks, logging and storing experimental logs. The benefit of writing the app in JavaScript is that the app can be easily updated by the backend without requiring the device to be connected via USB. With this approach, the latest JavaScript code is pulled as part of the web page and executed every time the benchmark is invoked.

Both the warmup and workload consist of running a CPU intensive task on all available CPU cores, for a fixed duration of time. In our experiments, the warmup phase was configured to run for 3 minutes to try and allow an idle CPU to heat up to the same state as a busy CPU. A busy CPU, on the other hand, would throttle and continue to maintain its heated state. This helps minimize the performance variance that can occur between the first experimental iteration and the subsequent iterations, as the first iteration normally had a cold start. We found that a warmup duration of 3 minutes was sufficient for obtaining consistent results. We chose a 5 minute duration for $T_{workload}$ to ensure that devices has ample time to heat up and exhibit any variations that may occur due to their thermal differences. The CPU intensive task consists of computing the digits of π in a loop on all available CPUs. Specifically, we compute the first 4,285 digits of π. This number was chosen as it was estimated to take roughly 1 second to compute at the

highest frequency on the Nexus 6. Performance is measured by the number of iterations the device is able to complete across all cores within $T_{workload}$.

When the intent is triggered, the app acquires a wakelock to ensure the device does not sleep and begins the CPU warmup phase. As soon as the CPU warmup is completed, the device releases the wakelock and starts the cooldown phase. In this phase, the device enters into a sleep state and wakes up momentarily every 5 seconds to poll the temperature sensor. As described earlier, this phase lasts until the temperature sensor reports a value that is below a pre-determined target temperature at which to start the workload. Figure 4 depicts the various events that occur during our ACCUBENCH technique.

Our experiments were performed on the Nexus 5, Nexus 6, Nexus 6P, LG G5 and Google Pixel handsets. The Nexus 5 and Nexus 6 ran Android 7.1 (Nougat) while the others ran LineageOS 15.1 Android 8.0 (Oreo). The reason for selecting LineageOS over stock Android were purely based on a simpler building and flashing experience.

To isolate the thermal effects of the CPU, Bluetooth, Radio and location services were disabled on every device. Additionally, the phone was locked thereby ensuring that the display was off during an experiment. In our custom-built version of the modular LineageOS, all apps that used Wi-Fi in the background were either disabled or removed entirely. This included auto-updates as well as all Google apps and services. Given the CPU-intensive nature of our workload, we are confident that the impact of other components such as DSP chips and memory controllers remained constant, if not negligible, throughout our experiments. Finally, we used the same enclosure for all experiments on each SoC. In other words, all Nexus 5 chips were tested within one Nexus 5 case and so on.

Since the operating system may alter device behavior based on battery conditions, we decided to eliminate this source of variance by powering our devices via the Monsoon Power Monitor [15]. We configured the Monsoon to provide the nominal voltage for each device as specified by the manufacturer.

As earlier studies such as [11] and [7] have shown, ambient temperature can play a crucial role in determining device performance. Following the best practices laid down by previous studies, all our experiments were performed in a controlled thermal environment which we refer to as THERMABOX. Temperature inside the THERMABOX was controlled using a RaspberryPi which measured the current temperature via a temperature probe. This RaspberryPi controller was also connected to a heating and cooling element which enabled it to regulate temperature within the THERMABOX. Heating and cooling the THERMABOX was achieved by power cycling a compressor and a 250W halogen lamp respectively. Figure 3 shows this setup.

We performed two sets of experiments on each chip, both using our ACCUBENCH technique. First, we allowed the CPU cores to run unconstrained—without frequency throttling—and measured performance. The underlying transistor variations would cause differences in leakage current which would in turn affect the temperature of the devices differently. These temperature differences meant that different chips throttled at different points thereby leading to performance variations. This workload is referred to as UNCONSTRAINED. Figure 4 depicts the temperature of the device as observed during an UNCONSTRAINED workload. Note how the CPU begins to throttle very quickly during the warmup and workload phases.

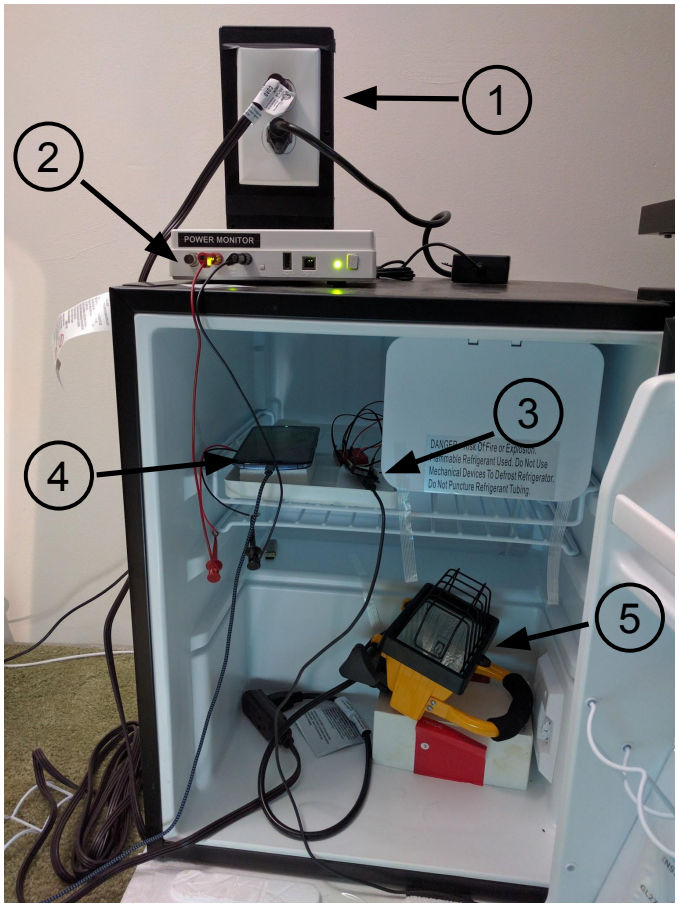


Fig. 3: **Controlled thermal environment**. All our experiments were run inside a controlled thermal environment with an ambient temperature of 26±0.5°C. 1) Temperature Controller (RaspberryPi), 2) Monsoon, 3) ESP-8266+Thermistor (Temperature Probe), 4) Device, 5) Heating Element.

Next, we constrained all CPU cores to run at a fixed, low frequency that was guaranteed to not thermally throttle. The goal of this experiment was to ensure that all chips performed the same amount of work which would then allow us to evaluate energy differences arising due to underlying transistor variations. We refer to this workload as FIXED-FREQUENCY, and Figure 5 describes this technique. Note that we are still running the workload for a fixed time duration instead of performing a fixed amount of work. While both approaches are equally susceptible to external influences such as background tasks, running the workload for a fixed duration gave us the additional advantage of being able to evaluate the reliability of our experimental setup in producing repeatable results, as we'd expect to see negligible performance variations.

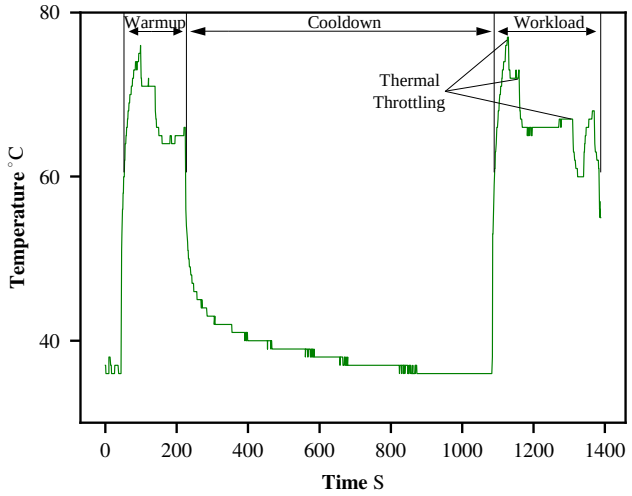


Fig. 4: **Various stages of ACCUBENCH during an UNCONSTRAINED workload on Nexus 5**. The warmup and cooldown phases together act as a mechanism to normalize device's thermal state.

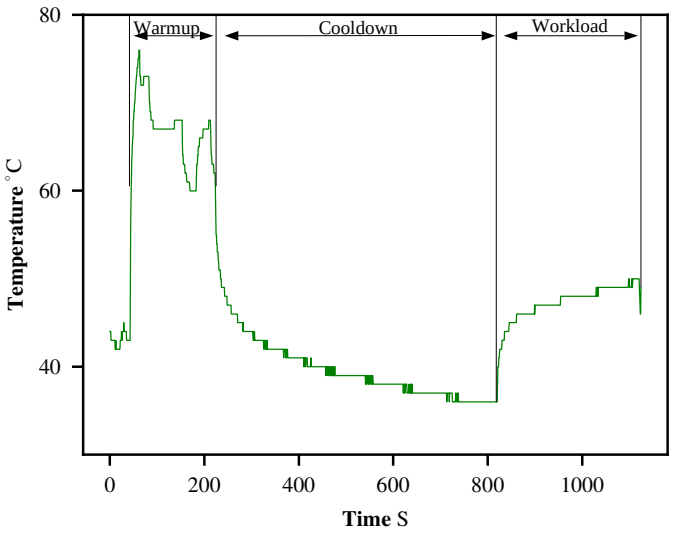


Fig. 5: **Thermal characteristics during FIXED-FREQUENCY workload on Nexus 5**. Due to a low frequency, the device never heats up to throttling levels.

Each workload was run a minimum of 5 times on each chip and we present the means with errors in all our results.

All experiments were run with a target ambient temperature of 26°C. The controller was configured to ensure that the temperature inside the THERMABOX always stayed within ±0.5°C of this target temperature. This setup was necessary to be able to produce reproducible results, particularly given how sensitive CPU performance can be to temperature.

Our first requirement as part of evaluating our ACCUBENCH technique was that the results be reproducible across multiple iterations. The entire process was automated by our benchmarking app which was able to communicate and configure the temperature controller and Monsoon power monitor. Upon firing a particular intent on the device, the app first communicates with the THERMABOX and confirms that it is within the target temperature range. Once stable, the app performs 5 iterations of our ACCUBENCH workload back-to-back. This process was repeated for each device.

IV. Experimental Results

This section is divided into three parts. First, we individually examine each SoC that was part of our study. Next, we discuss the source of performance variation in detail, and provide examples that illustrate the source of variation on multiple devices. Finally, we conclude with a summary and discussion of insights that we were able to glean from our study.

A. Individual SoCs

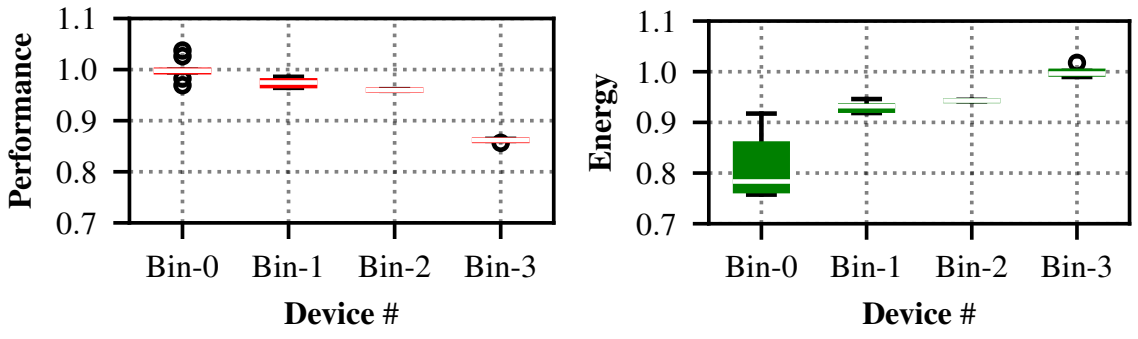
In each subsection, we describe the number of chips that were used as part of our study, any available CPU binning information that we could unearth, and the performance and energy consumption of the chips relative to one another. Throughout this section, we represent our results in a normalized form. This helps in depicting variations that occur between different chips. Errors are represented in the form of Relative Standard Deviation (RSD), or the absolute value of the coefficient of variation.

1) SD-800 & SD-805: The Snapdragon-800 SoC was released in January, 2013 on a 28nm process and has a quad-core Krait CPU designed by Qualcomm. The Snapdragon-805 was released later that year and featured a small increase in CPU frequency. We used the Nexus 5 smartphone to study the SD-800 and the Nexus 6 to study the SD-805. We used 4 Nexus 5 devices and 3 Nexus 6 devices in our study.

Both SoCs exposed their binning information and corresponding voltage-frequency tables at runtime. The tables for the Nexus 5 can be found in Table I. We were unable to find a similar voltage table that corresponded to CPU bins on the Nexus 6. We managed to obtain 5 out of the 7 possible bins for the SD-800. However, the bin-4 chip failed during our experiments and thus we are unable to provide results for this chip. We evaluate and discuss the result of the remaining 4 chips here. Results from the Nexus 6 are not presented here as they exhibited negligible performance (2%) and energy (2%) variations across all 3 devices.

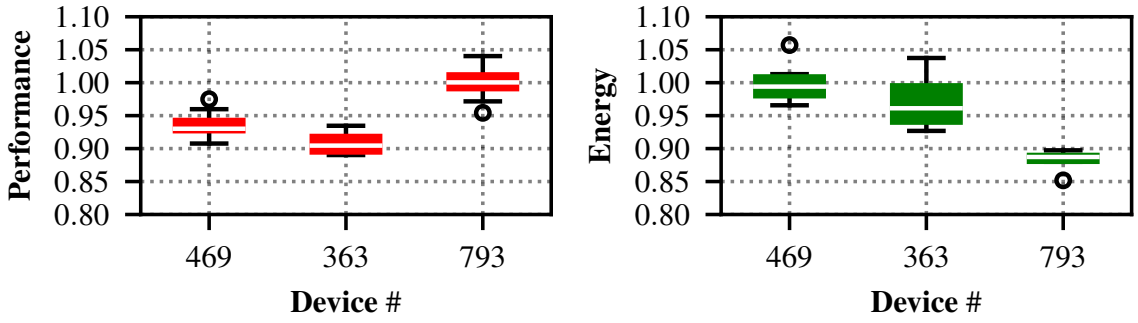
The SD-800 exhibits significant process variation which is described in Figures 6a, 6b. In our UNCONSTRAINED workload which is described in Figure 6a, bin-0 exhibits the highest performance while being 14% faster than bin-3, which exhibits the worst performance. The energy results from our FIXED-FREQUENCY workload are no different and are presented in Figure 6b. Again, bin-0 outperforms the other bins; it consumes 19% less energy than bin-3 to do the same amount of work. The performance variation for workload was within 1.3%.

Counterintuitively, despite having the highest operating voltage, bin-0 performs the best in terms of both performance and energy. This higher voltage has oftentimes been wrongly considered as a sign of being the worst bin [9]. The chips with the lower voltage were configured as such to reduce their inherently high leakage power.



(a) UNCONSTRAINED (b) FIXED-FREQUENCY

Fig. 6: **Process variations in SD-800 (Nexus 5)**. Process variations have significant impact with observed performance and energy variations of 14% and 19% respectively.



(a) UNCONSTRAINED (b) FIXED-FREQUENCY

Fig. 7: **Process variations in SD-810 (Nexus 6P)**. Performance and energy variations are in the order of 10% and 12% respectively.

2) SD-810: The Snapdragon-810 SoC was released in 2015 on a 20nm process, was built with ARM's big.LITTLE architecture and consisted of 8 CPU cores—4 big Cortex-A57 cores and 4 low-powered Cortex-A53 cores. The Nexus 6P used the SD-810 SoC and was launched in September 2015. We used 3 Nexus 6P devices as part of our study. All our devices reported being on 'speed-bin 0', and we were unable to determine the total number of bins present on this chipset.

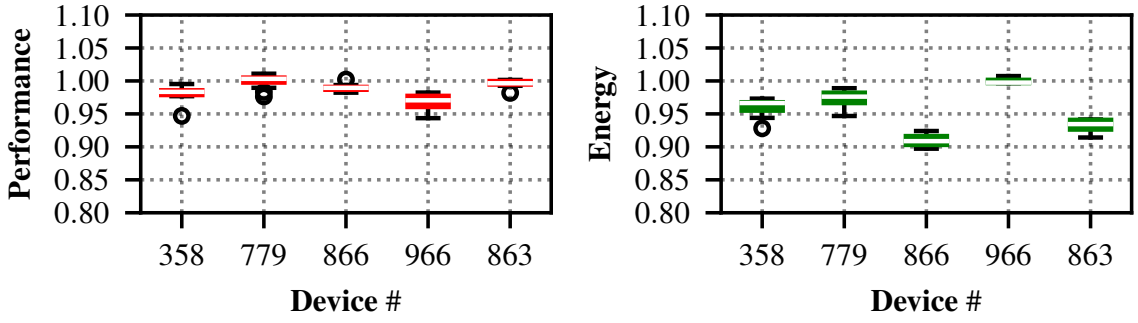
These devices along with other big.LITTLE cores implement a hardware block named Rapid-Bridge Core Power Reduction (RBCPR) [16] [17] that provides a feedback loop to optimize the voltage settings for each core. These runtime voltage settings are determined based on the binning process and current temperature of the chip. Thus, it is likely that there is no static voltage-frequency table to extract from the kernel sources. Figures 7a and 7b report the results of our Nexus 6P study. Device-363 exhibited 10% lower performance while consuming 12% additional energy when compared to device-793. Performance variations during the FIXED-FREQUENCY workload was computed to be RSD 2.63%.

3) SD-820 & SD-821: We discuss the SD-820 and SD-821 chips side-by-side as they share similar characteristics.

The Snapdragon-820 had its first phones released in early 2016 while the Snapdragon-821 debuted in late 2016. Both feature a process upgrade to 14nm FinFET and consist of a quad-core Kryo CPU—a reduction in core count from the SD-810's octa-core CPU possibly due to the significant levels of thermal throttling on the Nexus 6P [18]. We used the LG G5 which was released in April, 2016 to study the behavior of the Snapdragon-820 and the Google Pixel to study the Snapdragon-821. Unlike it's predecessors, both chips exposed neither binning information nor voltage tables.

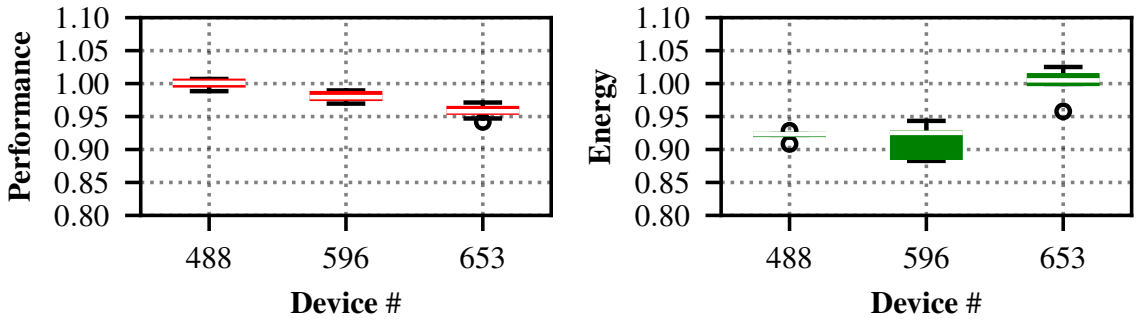
The LG G5 was different in that it also exhibited the unique characteristic of throttling based on battery voltage. As described in Section III, we configured the Monsoon to provide the nominal voltage that was listed on each device's battery. In the case of the LG G5, this was 3.85V. However, when comparing results obtained from the Monsoon and the battery, we found that all the results from the Monsoon performed significantly worse than those from the battery. Further investigation revealed that the OS was throttling the device based on input voltage. By configuring the Monsoon to provide the maximum voltage of 4.4V as listed on the battery, we were able to obtain performance on par with the battery. These results are shown in Figure 10.

Figures 8a, 8b describe the performance and energy variations of the SD-820. While Figures 9a, 9b describe the SD-821. Both chips exhibit similar characteristics by exhibiting performance variations of ≈5% and energy variations of the order of ≈ 10%. Although the performance variations that we observed were minimal, we are confident that these are real variations with our errors being 1.2% for the LG G5 and 0.7% for the Google Pixel.



(a) UNCONSTRAINED (b) FIXED-FREQUENCY

Fig. 8: **Process variations in SD-820 (LG G5)**. Exhibits low performance variations of 4% but energy variations of 10%.



(a) UNCONSTRAINED (b) FIXED-FREQUENCY

Fig. 9: **Energy variations in SD-821 (Google Pixel)**. Very similar behavior to the SD-820. Performance and energy variations in the order of 5% and 9% respectively.

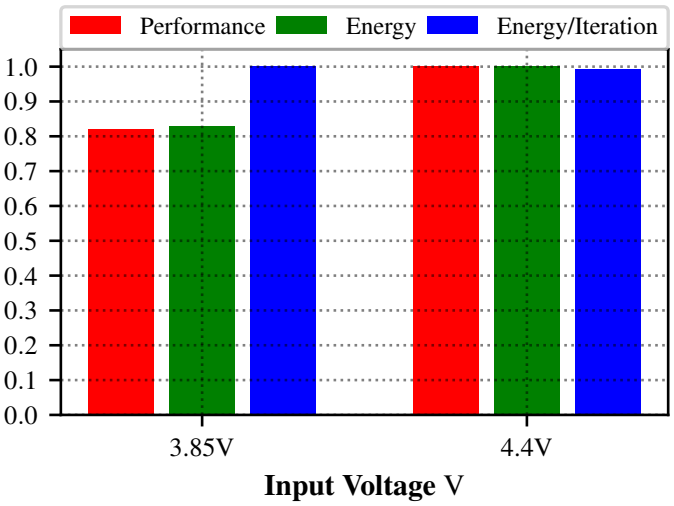


Fig. 10: **LG G5 anomalous behavior**. The LG G5 behaves throttled when the input voltage is set to the nominal battery voltage of 3.85V.

B. Source of Performance Variation

In Section II we introduced how underlying transistor variations cause performance variations to occur. Here, we depict the phenomenon in action and provide a detailed explanation of the same. Figure 11 shows the distribution of the observed temperatures and frequencies of two iterations performed on two different Google Pixel smartphones. In these iterations, device-488 exhibited 7% higher performance than device-653 and that device-488 also had a 2% and 7% higher frequency on average for CPUs 0 and 2 respectively.

For the Pixel, the temperature data is perhaps counterintuitive. Figure 11 shows device-488 spending more time at higher temperatures than device-653 which should imply that device-488 gets throttled more, but this is not the case. On investigating the results, we found that due to the transistor characteristics and the thermal throttling policy of the Google Pixel, device-653 gets throttled harder as its temperature did not drop as drastically as device-488 upon initially being throttled. So, time spent at temperature is not sufficient to capture the complexities of thermal throttling mechanisms.

We saw similar behavior in terms of frequency and temperature distributions for the Nexus 5. When comparing an iteration from a bin-1 Nexus 5 and a bin-3 Nexus 5, we saw bin-1 outperform bin-3 by 11%. Figure 12 shows that for these experiments, the bin-1 device ran at higher frequencies, with the mean frequency also 11% higher.

These results confirm our claim that the differences observed in our results are due to thermal throttling and not due to external activity such as background tasks. For devices of the same model, experiments yield consistently lower performance which is caused by the device running at lower frequencies due to different thermal throttling behavior. This behavior is caused by differences in the SoC.

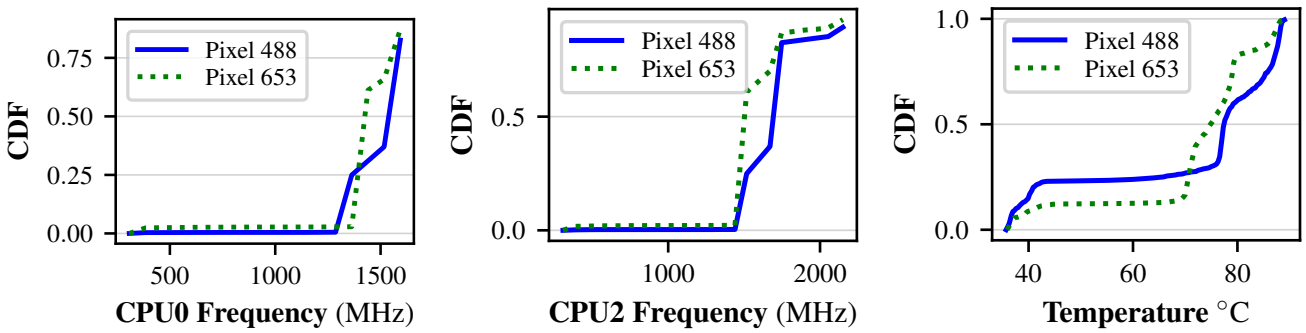


Fig. 11: **Frequency and temperature distributions over time for Pixel experiments**. Mean frequency varies by 7% and matches the observed performance variations.

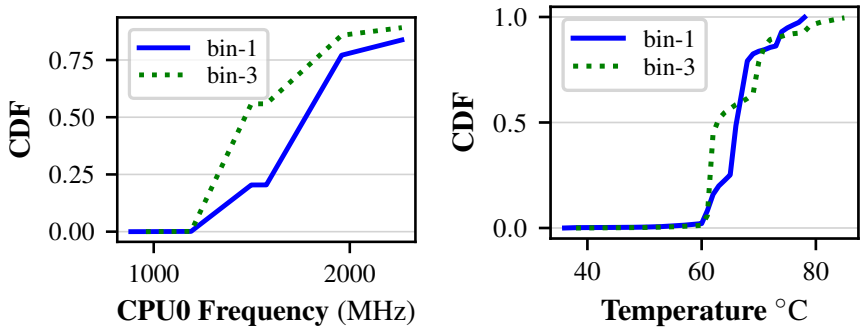


Fig. 12: **Frequency and temperature distributions over time for Nexus 5 experiments**. Mean frequency varies by 11% and matches the observed performance variations.

Chipset	Model	# Devices	Variation (%)	
			Performance	Energy
SD-800	Nexus 5	4	14%	19%
SD-805	Nexus 6	3	2%	2%
SD-810	Nexus 6P	3	10%	12%
SD-820	LG G5	5	4%	10%
SD-821	Google Pixel	3	5%	9%

TABLE II: **Summary of energy-performance variations**.

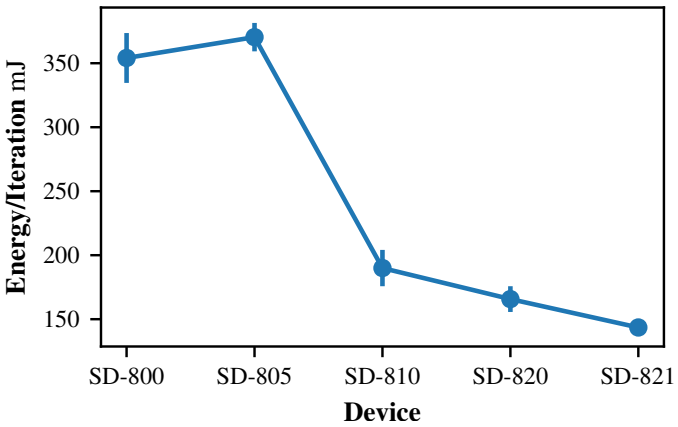


Fig. 13: **Relative efficiency of various smartphone SoCs**. While the SD-805 is definitely more performant than the SD-800, it comes at the cost of decreased efficiency.

C. Summary & Discussion

In this section, we summarize our results in brief and offer some discussion on our results, their implications and impact.

- **Variations continue to exist on newer chips and affect smartpone performance**. Process variations were known to exist and affect smartphone performance on older chipsets, such as the SD800 [7]. Our results show that they continue to exist in the newer chips and at times with differences in the range of 10% energy variations. Table II summarizes these performance and energy variations.
- **Quantifying efficiency improvements across SoC generations**. Another interesting dimension that we were able to explore was efficiency. While manufacturers announce new SoCs by touting their performance and energy improvements over the previous generation, we were unable to find any sources depicting efficiencies. These results are described in Figure 13. Although efficiency has improved as a whole with improving manufacturing process and reducing transistor sizes, in our experiments, the SD-805 was, on average, found to be less efficient than its predecessor SD-800.
- **Complications of Non-Thermal Throttling**. The strange behavior of the LG G5 wherein the CPU was being throttled by $\approx$ 20% based on input voltage is reminiscent of recent reports of old iPhones being throttled [19]. The voltage that a battery is able to supply decreases over time and throttling based on the input voltage deteriorates user-perceived performance and complicates benchmarking as researchers have to now account for more than just the battery capacity.
- **Smartphone Binning & Ranking**. Battery life is a universal concern among smartphone users. Given a choice, consumers will undoubtedly gravitate towards a smartphone that lasts longer. However, currently, this information is not made available to them. Smartphone reviewers must make

it a point to review more than just one device and include energy-performance variations as part of their reviews.

V. Related Work

Characterizing smartphone behavior in terms of temperature, energy, and performance has been attempted by numerous academic and industry researchers. However, to the best of our knowledge, we are the first to attempt to use these characteristics to determine the quality of the underlying silicon.

Sekar describes the challenges faced in thermal aware power management of mobile devices [20]. The paper discusses how power management policies are typically temperature agnostic, despite temperature having a significant impact on leakage and dynamic power. Other works such as [21] and [22] characterize the impact of power dissipation on different aspects of user experience such as skin temperature and performance. Halpern et al. takes this one step further and also attempts to quantify user satisfaction [23]. They don't, however, consider the effects of ambient temperature and don't quantify process variations.

Multiple efforts have also modeled thermal behavior of smartphones. Lee et al. developed three dimensional finite element thermal models using the size and power dissipation data of commercial hand-held devices [24]. Therminator is a full device thermal analyzer for smartphones that is capable of generating accurate temperature maps for chip containing multiple hardware components and the skin of the device [25]. There have also been several efforts to model and measure energy consumption of CPUs on mobile devices. For example, Nachiappan et al. present a multi-component energy management of mobile devices for frame-based applications [26]. They use simulation models that use multiple activity counters to estimate energy consumption of cores. Lee et al. present an energy management approach for mobile interactive web workloads to maintain cloud-guided QoS [27]. They measure energy consumption of Cortex-A7 and Cortex-A15 cores using onboard energy sensors on an ODROID-XU3 board. We record the power consumption of mobile devices under test by measuring the current drawn by these devices using Monsoon, an external power monitor.

VI. Future Work

CPU binning information is something that has been kept secret for the most part by manufacturers and despite their best efforts at attempting to level the playing field between chips belonging to different bins, the underlying transistor variations and its effects are inescapable. We strongly believe that device binning information including the number of bins and what they mean would greatly help smartphone consumers.

The best way to obtain this information in the absence of manufacturer's assistance, would be to introduce a benchmarking app on Google Play with the express intent of gathering the necessary data for binning CPUs. We plan to build such an app and evaluate its efficacy in the future. The only parameters that we cannot control for in the wild are ambient temperature and software stack. However, preliminary results on using the cooldown phase as an estimate of ambient temperature are encouraging. This, in addition to strict filters, should enable us to compare different devices from across the world.

Our goal would be to gather sufficient data from devices of various smartphone models via crowdsourcing and then using this data to rank other devices, thereby helping users and researchers determine the characteristics of their smartphone and how it compares to other smartphones of the same model. Not only can the devices be ranked on the absolute scale with respect to one another, but the gathered information can also be used to understand how the manufacturers are binning their CPUs and the distribution of various bins. In cases where there is no clear bin labels as observed on the Nexus 5 and Nexus 6, we plan to create our own bins by clustering the performance data using unstructured learning algorithms.

VII. Conclusion

In this work, we described a problem that currently plagues smartphone consumers and researchers: differences in smartphone CPUs affect both energy and performance. We introduced our technique and careful methodology of identifying underlying transistor quality which we then used to quantify energy and performance variations in 5 different SoC generations. Our contributions are summarized below:

- **Introduce methodology for reliable energy-performance measurements**. We strongly believe that our experimental setup is a contribution unto itself with an average error of 1.1% RSD over roughly 300 iterations of our workloads. Researchers seeking to accurately measure energy or performance characteristics are encouraged to replicate our experimental setup.
- **Quantify process variation on smartphones**. By no means is process variation new. It has been studied extensively in the past over the last few decades. However, to the best of our knowledge, process variation on smartphones has never been quantified. We showed that process variation continues to exist across SoC generations and even recent SoCs exhibit energy and performance variations in the order of 5% and 10% respectively.
- **Provide lower-bound on energy and performance variations**. It only takes two devices to observe variations. While our study of SoCs is limited, at times with only 3 devices to represent an SoC generation, the process variations shown in Table II can be considered as a minimum lower-bound to the overall variation for each SoC. In other words, a larger study may unearth that the full extent of energy variations on the SD-820 to be greater than 9%, however, our work establishes that the full extent is at least 9%.

VIII. Acknowledgements

This material is based on work partially supported by NSF Awards CSR-1409014 and CSR-1409367. Any opinion, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

REFERENCES

[1] O. S. Unsal, J. W. Tschanz, K. Bowman, V. De, X. Vera, A. Gonzalez, and O. Ergin, "Impact of parameter variations on circuits and microarchitecture," IEEE Micro, vol. 26, no. 6, pp. 30–39, Nov 2006.

[2] S. Borkar, T. Karnik, S. Narendra, J. Tschanz, A. Keshavarzi, and V. De, "Parameter variations and impact on circuits and microarchitecture," in Proceedings 2003. Design Automation Conference (IEEE Cat. No.03CH37451), June 2003, pp. 338–342.

[3] K. A. Bowman, S. G. Duvall, and J. D. Meindl, "Impact of die-to-die and within-die parameter fluctuations on the maximum clock frequency distribution for gigascale integration," IEEE Journal of Solid-State Circuits, vol. 37, no. 2, pp. 183–190, Feb 2002.

[4] L. Zhang, L. S. Bai, R. P. Dick, L. Shang, and R. Joseph, "Process variation characterization of chip-level multiprocessors," in 2009 46th ACM/IEEE Design Automation Conference, July 2009, pp. 694–697.

[5] S. Dighe, S. R. Vangal, P. Aseron, S. Kumar, T. Jacob, K. A. Bowman, J. Howard, J. Tschanz, V. Erraguntla, N. Borkar et al., "Within-die variation-aware dynamic-voltage-frequency-scaling with optimal core allocation and thread hopping for the 80-core teraflops processor," IEEE Journal of Solid-State Circuits, vol. 46, no. 1, pp. 184–193, 2011.

[6] B. Raghunathan, Y. Turakhia, S. Garg, and D. Marculescu, "Cherry-picking: exploiting process variations in dark-silicon homogeneous chip multi-processors," in Design, Automation & Test in Europe Conference & Exhibition (DATE), 2013. IEEE, 2013, pp. 39–44.

[7] G. P. Srinivasa, R. Begum, S. Haseley, M. Hempstead, and G. Challen, "Separated by birth: Hidden differences between seemingly-identical smartphone cpus," in Proceedings of the 18th International Workshop on Mobile Computing Systems and Applications. ACM, 2017, pp. 103–108.

[8] V. Zolotov, C. Visweswariah, and J. Xiong, "Voltage binning under process variation," in Proceedings of the 2009 International Conference on Computer-Aided Design. ACM, 2009, pp. 425–432.

[9] "Redefining the android experience with google's nexus 5," https://goo.gl/1yaiS6.

[10] "Ranking - antutu benchmark," http://www.antutu.com/en/ranking/rank1.htm.

[11] Y. Guo, Y. Xu, and X. Chen, "Freeze it if you can: Challenges and future directions in benchmarking smartphone performance," in Proceedings of the 18th International Workshop on Mobile Computing Systems and Applications, ser. HotMobile '17. New York, NY, USA: ACM, 2017, pp. 25–30. [Online]. Available: http://doi.acm.org/10.1145/3032970.3032979

[12] "Antutu benchmark," https://play.google.com/store/apps/details?id=com.antutu.ABenchMark&hl=en.

[13] "Lineageos statistics," https://stats.lineageos.org.

[14] N. S. Kim, T. Austin, D. Baauw, T. Mudge, K. Flautner, J. S. Hu, M. J. Irwin, M. Kandemir, and V. Narayanan, "Leakage current: Moore's law meets static power," computer, vol. 36, no. 12, pp. 68–75, 2003.

[15] "Monsoon power monitor," https://goo.gl/rlizsj.

[16] "Qualcomm snapdragon 410e processor apq8016e system power overview," https://developer.qualcomm.com/qfile/35136/lm80-p0436-73_a_qualcomm_snapdragon_410e_processor_apq8016e_system_power_overview.pdf.

[17] "Linux kernel cpr-regulator," https://android.googlesource.com/kernel/msm/+/android-8.1.0_r0.112/Documentation/devicetree/bindings/regulator/cpr-regulator.txt.

[18] "In-depth with the snapdragon 810s heat problems," https://arstechnica.com/gadgets/2015/04/in-depth-with-the-snapdragon-810s-heat-problems/.

[19] "Oh apple, you really need to rethink how you do things," https://www.zdnet.com/article/oh-apple-you-reeally-need-to-rethink-how-you-do-things/.

[20] K. Sekar, "Power and thermal challenges in mobile devices," in Proceedings of the 19th annual international conference on Mobile computing & networking. ACM, 2013, pp. 363–368.

[21] A. Straume, G. Oftedal, and A. Johnsson, "Skin temperature increase caused by a mobile phone: a methodological infrared camera study," Bioelectromagnetics, vol. 26, no. 6, pp. 510–519, 2005.

[22] P. Mercati, T. S. Rosing, V. Hanumaiah, J. Kulkarni, and S. Bloch, "User-centric joint power and thermal management for smartphones," in Mobile Computing, Applications and Services (MobiCASE), 2014 6th International Conference on. IEEE, 2014, pp. 98–105.

[23] M. Halpern, Y. Zhu, and V. J. Reddi, "Mobile cpu's rise to power: Quantifying the impact of generational mobile cpu design trends on performance, energy, and user satisfaction," in High Performance Computer Architecture (HPCA), 2016 IEEE International Symposium on. IEEE, 2016, pp. 64–76.

[24] J. Lee, D. W. Gerlach, and Y. K. Joshi, "Parametric thermal modeling of heat transfer in handheld electronic devices," in Thermal and Thermomechanical Phenomena in Electronic Systems, 2008. ITHERM 2008. 11th Intersociety Conference on. IEEE, 2008, pp. 604–609.

[25] Q. Xie, M. J. Dousti, and M. Pedram, "Therminator: a thermal simulator for smartphones producing accurate chip and skin temperature maps," in Low Power Electronics and Design (ISLPED), 2014 IEEE/ACM International Symposium on. IEEE, 2014, pp. 117–122.

[26] N. C. Nachiappan, P. Yedlapalli, N. Soundararajan, A. Sivasubramaniam, M. T. Kandemir, R. Iyer, and C. R. Das, "Domain knowledge based energy management in handhelds," in High Performance Computer Architecture (HPCA), 2015 IEEE 21st International Symposium on. IEEE, 2015, pp. 150–160.

[27] W. Lee, D. Sunwoo, A. Gerstlauer, and L. K. John, "Cloud-guided qos and energy management for mobile interactive web applications," in Mobile Software Engineering and Systems (MOBILESoft)., 2017.