Reap What You Store: Side-channel Resilient Computing Through Energy Harvesting

Michael Moukarzel*
Worcester Polytechnic Institute
mamoukar@wpi.edu

Matthew Hicks*
Virginia Tech
mdhicks2@vt.edu

ABSTRACT

A hidden dimension of software and hardware security is secret-revealing information disseminated through side channels. Even the most secure systems tend to reveal their secrets through secret-dependent computation. Secret-dependent computation is detectable by monitoring a system's time, power, outputs, and electromagnetic signature. Common defenses to side channel emanations include adding noise to the channel or making algorithmic changes to eliminate specific side channels. Unfortunately, existing solutions are either, not automatic, not comprehensive, and/or not practical.

We propose an isolation-based approach for eliminating power and timing side-channels that is automatic, comprehensive, and practical. Our approach eliminates side channels by leveraging energy harvesting techniques to isolate trusted computation from the rest of the system. Software has the ability to request a fixed-power and fixed-time quantum of isolated computation. By discretizing power and time, our approach controls the granularity of side channel leakage; the only burden on programmers is to ensure that all secret-dependent execution differences converge within a single power/time quantum.

We design and implement three approaches to power/time-based quantization and isolation: a wholly-digital version, a hybrid version that uses capacitors for time tracking, and a full-custom version. A key insight we leverage is that capacitors act as resource efficient, workload and environment independent time trackers. We explore the trade-offs of the three designs and look at the challenges ahead.

CCS CONCEPTS

• **Security and privacy** → **Security in hardware**; • **Hardware** → **Energy generation and storage**;

KEYWORDS

Side-channels, Energy harvesting, Intermittent computation

*Portions of this work completed at MIT Lincoln Laboratory.

ENSsys'17, November 5, 2017, Delft, Netherlands

ACM ISBN 978-1-4503-5477-6/17/11.
https://doi.org/10.1145/3142992.3142996

ACM Reference Format:
Michael Moukarzel and Matthew Hicks*. 2017. Reap What You Store: Side-channel Resilient Computing Through Energy Harvesting. In *Proceedings of ENSsys'17:The Fifth ACM International Workshop on Energy Harvesting and Energy-Neutral Sensing Systems , Delft, Netherlands, November 5, 2017 (ENSsys'17),* 6 pages.
https://doi.org/10.1145/3142992.3142996

1 INTRODUCTION

Side-channel emanations compromise secure execution by revealing sensitive data. There is a wide range of channels that can be used to extract secure information, but power [3] and timing [4] side-channels are the two most readily exploited by attackers. All electronic devices generate power and timing information during execution. This information is observable by monitoring the power rail—using relatively common equipment—to form traces. Attackers analyze execution traces or differences between traces to deduce (or extract directly) otherwise secret information. The root cause is secret-dependent emanation.

Existing countermeasures include: reducing the magnitude of side-channel emanations, the addition of noise to mask side-channel emanations, obfuscation to hide the relative timing of the secret-revealing emanation, and incorporating randomness at the software level [7]. Alternatively, we advocate isolation; our countermeasure works by harvesting resources and then performing computation with secret-revealing side-channels in isolation from the rest of the system—and the attacker. By harvesting the resources needed for secure execution and then performing the secure execution in isolation, our approach effectively quantizes the information revealed by power and timing side channels. Assuming no secret-dependent execution-difference spans quantized executions, then, from the attacker's perspective, the power and timing requirements are uniform and secret-independent. The challenge is to design a harvesting and isolation system where we eliminate both power and timing side channels.[1] We achieve this using a quantization controller: in-package control logic that sits between a secure processor and the rest of the system. The quantization controller dictates where the secure processor gets its power; either from the system's power rail or from an energy storage capacitor (common to energy harvesting systems). The quantization controller also has counters (either digital or analog) to ensure that secure execution time is execution-invariant.

[1] Many techniques for eliminating power side-channels result in increasing the information in timing side-channels and vice versa. Given that there is *no* added burden for an attacker capable of measuring power to measure time—and timing side-channels have less noise—it is not acceptable for a defense to only address one, but not the other.

Lastly, to provide resilience to thermal-based fault attacks, we leverage a DINO-like [1, 5] approach to ensure atomicity of each secure computation.

To demonstrate our approach and explore its trade-off space, we implement three quantization controllers: a wholly-digital, a hybrid, and a full-custom controller. For the hybrid and full-custom controller, we use capacitors and their natural leakage as execution-independent counters. We simulate each controller design using a mix of digital and analog simulation. Our results show that all three controllers effective quantize power and time to eliminate their respective side channels. Our results also indicate that the hybrid design is optimal in terms of ease of implementation combined with compactness.

This paper presents the following contributions:

- We are the first to use energy harvesting techniques to increase security: we leverage techniques from energy harvesting systems to quantize a chip's power consumption and isolate it from the rest of the system. This eliminates fine-grain, execution-based, externally-visible power side-channels.
- We use execution-independent counters to quantize a chip's execution time. This eliminates fine-grain, execution-based, externally-visible timing side-channels.
- We are the first to make side-channel protection a dynamic and software-level decision.
- We design three power/time quantization controllers—two that use the natural leakage of capacitors to act as compact and temperature-fault-attack immune counters.

2 BACKGROUND

All physical devices have side-channel emanations; this is a byproduct of physical implementation and computation. The key to side-channel analysis is the interpretation of these leaks to reveal secret information. As such, there are many different side channel leaks used to capture secrets. These include: power, timing, electromagnetic, acoustic, memory remanence, and thermal. The two most commonly exploited side channels are power (trough differential power analysis) and timing. Thus, the following threats are the focus of this paper:

- **Power Analysis**: Statistical analysis of the power rail during execution.
- **Timing Analysis**: Analysis of execution time and performance.
- **Electromagnetic (EM) Analysis**: Analysis of the EM generated by the secure chip.
- **Fault Attacks**: Introducing glitches through temperature manipulation or unexpected power transitions.

Many commercially available energy harvesting devices are used in a variety of secure executions; for example, the square point-of-sale credit card reader. As such, sensitive information, e.g., cryptography keys, need to be kept secure from leakage during execution. The first step in designing any countermeasures is to assess what an attacker would be capable of and information he has access to. An attacker performing any kind of power side channel analysis would need to have physical access to the power rail. This is particularly easy to achieve with low power embedded devices. Having access to the power rail not only provides access to the power rail, but also provides insight into timing. As such, our threat model includes an attacker with full physical access to any information outside the secure processor's package.

There is a symbiotic relationship between the different side channels:

- **Power vs Timing**: Implementing any limitations on a power rail would directly correlate to any variability in execution performance. Therefore a leakage hidden on the Power rail would be exposed through timing analysis.
- **Timing vs Power**: Implementing any timing limitations to hide execution would directly translate into power consumption. Execution that was being obfuscated in constant execution times would be visible in the variable power draw.
- **Timing vs Fault**: Implementing counters to conceal timing differences creates opportunities for fault attacks that invert the expected ordering of system events. Temperature-based fault attacks effect the rate of leakage of capacitor while not effecting digital counters.
- **Power vs Electromagnetic**: Hiding the power capacitor recharge side channel requires clamping the capacitor's voltage down to a fixed level. This creates a large current spike visible in the electromagnetic spectrum. The magnitude of this current spike indicates how much current was on the power capacitor after secure execution, thus revealing secret information.

Without care, eliminating the leak from one side channel shifts the information to another channel. For example, in our energy harvesting based approach, without discharging a capacitor to a fixed level, the time required to completely charge the power capacitor indicates how much energy was required by secure execution. This increases timing side channel information. A complete solution addresses the following threats in unison:

The goal of this paper is to design a defense that addresses both power and timing side channel attacks. We also consider the effects of power faults in the implementation of our design. In future work, we will add defenses to electromagnetic side channel attacks and additional defenses to fault attacks.

3 DESIGN

Our goal is to develop a side-channel countermeasure that addresses both power and timing leakage sources. Addressing both side channels simultaneously is both challenging and necessary because many defenses push leakage from one side channel to another, instead of eliminating the leakage. Given our threat model of an attacker able to probe pins on the trusted chip's package, it is expected that the attacker is able to accurately measure voltage, current, and the time associated with externally visible events.

Our proposed countermeasure stems from the observation that existing energy harvesting systems effectively isolate the chip they power from the source of power. By inserting switches along with the support for energy harvesting, it is possible to forcefully disconnect a secure processor from a continuous power source. Specifically the evolution of ultra-low-power embedded devices has encouraged a transition away from bulky, expensive, and flammable batteries to capacitor banks. The transition was hastened by the development of energy harvesting techniques, allowing a more rapid recharge of the

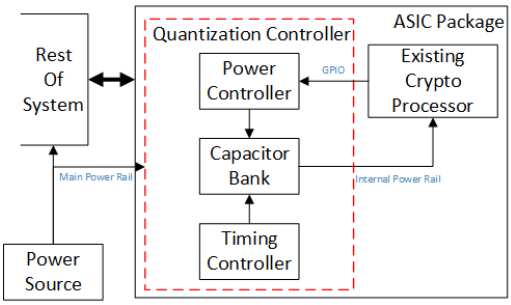

Figure 1: *System Block Diagram*

power capacitor. These power capacitors are charged using a variety of different energy harvesting techniques, allowing for truly isolated low power embedded devices. μLeech is a lower power embedded device that utilizes such techniques [6]. It has the ability to execute encryption schemes using the power siphoned from a smart-phone's audio port. Although most phones cannot provide sufficient power directly, μLeech contains a large power capacitor and charge pump capable of providing sufficient power for one round of the AES-128 block cipher. The authors report that one round of AES, encryption or decryption, consumes 555μA and takes 140μs at 1MHz [6] (for their outdated micro-controller).

These energy harvesting technique act as an isolating mechanism, removing the reliance on a continuous power source. Strategically creating power discontinuities to quantize power into execution-independent sized chunks (called a quantum) acts as a security mechanism. While executing (insecurely) on the continuous power supplied by the power rail, the power capacitor is charged so that it it can provide sufficient power during isolated, secure execution. Using this power capacitor and our proposed quantization controller we design a chip with a separate internal isolate-able power rail (Figure 1). The internal power rail isolation is triggered by the secure processor (or a dedicated cryptographic core) when it is ready to start secure execution. This allows the software to dynamically balance security guarantees and performance. As a result, any feedback that is generated on the power rail and used for power side-channel analysis is now isolated from the main power rail. Therefore an attacker looking for power leakage on the main power rail would be unable to see the feedback generated on the internal power rail during execution. Isolating the internal power rail allows us to eliminate power side channels.

Although isolating the power rail eliminates some forms of power leakage, it also creates new timing and power leaks. One new source of side channel information is the amount of time required for secure execution; an attacker measures this as the amount of time the chip is isolated from the power rail. A second new source of side channel information is in the amount of time required to recharge the power capacitor. This stems from the fact that the time required to charge a capacitor depends on the capacitor's charge at the start of charging. In cases where secret-dependent execution results in a different final charge remaining on the power capacitor, the amount of time it takes to charge the capacitor indicates that starting charge. A third new source of side channel leakage, related to the second, is that an attacker can measure the voltage at a chip's power pins at the start of recharge to check for differences in secure execution total power

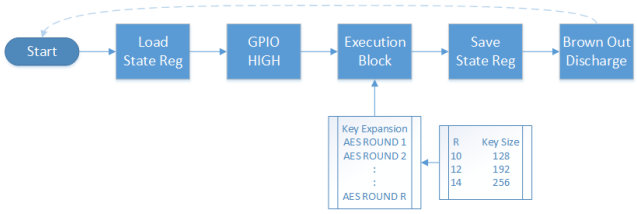

Figure 2: *AES-128 Flow Chart*

consumption. Our quantization controller design eliminates all three of these new side channel sources.

To address all potential power and timing side channels, we quantize both power and time into secret-independent chunks that we fit secure execution into. Generating uniform power and time per execution requires us to predefined an execution cycle, for this we turn to low-power intermittent processing techniques. These techniques allow us to subdivide long running computations into shorter tasks while guaranteeing forward progress at the granularity of a power/time quantum, as shown in previous literature [1, 2, 5]. Clank is lightweight architectural support that aides in the execution of a long-running application on harvested energy with intermittent computation. A processor with Clank architectural support is able to divide any secure execution into sub-divided secure executions that can be isolated individually. Alternatively using the Dino and Chain tools a secure execution can be subdivided into programmer-defined tasks. Dino (Death Is Not an Option) allows programmers to set task boundaries to subdivide long-running computations into shorter tasks. While Chain would be used to guarantee forward progress at task granularity. These tools allow us to perform any secure execution as intermittent executions, however intermittent secure execution does not create uniform power and time. As long as the secure execution dictates the time or power per execution there will not be uniformity on either channel. To isolate our secure sub-execution from leakage the power and time per execution cycle need to be independent of the secure sub-execution.

We would effectively be quantizing our execution per capacitor charge cycle with a timing controller. In essence, instead of executing per cycle, we would be executing per quantum. Executing would be limited to the available power and time per quantum, obfuscating any secure execution per quantum. With intermittent execution, we can guarantee sub executions within our prefixed execution time. Using a timing controller, Figure 1, we eliminate time per execution leakage be enforcing uniform time per quantum. However, we would still be leaking power per execution based on the recharge rate and time of our capacitor bank between quantums, allowing an attacker to deduce information per qaunt execution. We have uniform timing per quantum but we also need uniform power per quantum. Our solution was to implement a power controller, Figure 1, to full discharge of our power capacitor before the end of each qaunt. Thereby guaranteeing uniform power and time per quantum.

In our initial design, we choose to implement AES-128 as a proof of concept. AES is a commonly used cryptographic scheme that is naturally quantizable and can be manually implemented as intermittent execution, as shown in Figure 2. AES is executed in serial rounds, meaning an AES round can be executed per quantum. Therefore our power capacitor is designed to last for one round of AES

with enough supplementary power to provide a save and load state before the processor hits brown out. Using the measured results from μleech we have our minimum time and power requirements for one round of AES: 555μA for 140μs at 1MHz on an AT32UC3L064 [6]. Although any algorithm can be implemented using Dino and Chain to meet the same quantum time and power limits. Previous literary work has shown that AES is susceptible to DPA attacks [3], using measurements from a single round, making AES an ideal choice to test and validate our countermeasures. Our proposed quantization controller, Figure 1, will control the secure execution, discharge, and recharge times of our power capacitor to generate uniform power and time leakage per AES round execution. The following is a list of events controlled by the quantization controller:

(1) **Secure Mode**:
- Initiated by cryptographic core via internal GPIO
- Internal power rail isolated from main power rail
- Power and Timing Counters initiated
- Secure Execution Initiated

(2) **Discharge Mode**:
- Power counter triggers power capacitor full discharge

(3) **Recharge Mode**:
- Timing counter triggers power capacitor recharge
- Main power rail reconnected

With the discharge and recharge modes, the quantization controller can guarantee uniform power and time per execution. Provided that each secure execution can be executed within the time and power allotted per quantum, there is no correlation between the secure execution and uniform power and time footprint. As the footprint is determined by the power and timing control counters: the power counter will guarantee that each execution appears to utilize the same power, and the timing counter will guarantee that each execution appears to take the same time. These counters could be exploited to create leakage through fault based side channel attacks. We were especially concerned with thermal based fault attacks that would cause a desynchronization between the counters and power capacitor, resulting in a loss of power before our counters reach their triggers. Our proposed design uses capacitor based counters instead of digital based counters mitigating thermal based fault leakage.

4 IMPLEMENTATION

Using existing software techniques it is possible to simulate similar power consumption per quantum, obfuscating the secure execution leakage. Our proposed design is a hardware quantization controller that will cause the power and timing signature to be uniform per quantum execution isolating any secure execution leakage. The proposed quantization controller, illustrated in Figure 1, would be between the main power rail and integrated within the package of a chip. Allow our quantization controller to create an isolated internal power rail with a power capacitor. Integrating this within the chip's EM shielded package prevents our power isolation from being circumvented and removes any EM leakage generated from our design. During the design of our quantization controller, we developed three different versions, each resistant to power and timing leakage but with their own advantages and disadvantages.

Initially, we designed a fully digital quantization controller using flip-flops and registers, illustrated in Figure 3. As described in our design section this quantization controller creates a uniform power and timing footprint per quantum. This is achieved by controlling the *Power Capacitor* per quantum through three modes: Secure, Discharge, and Recharge. To enter Secure Mode, the *Processor* will trigger our controller with a *GPIO*. This will trigger a *Power Switch* to isolate the *power capacitor* from the *power source*, allowing for secure execution using the *power capacitor* as the source. During this execution, any feedback leakage on the power rail will be isolated to the *internal power rail*. The *power capacitor* will provide sufficient power for the full quantum execution. However, the next mode is not triggered by the completion of the quantum execution. Discharge Mode is reached when the *Discharge Counter* triggers the *Discharge Switch* that will short the remaining power in the *power capacitor*. Recharge Mode is then reached when a longer *Recharge Counter* triggers the *Recharge Switch* that causes the *power switch* to reconnect the *power source* to the *power capacitor*. Ending the quantum cycle until the processor is ready to initiate the next quantum execution.

Table 1: *Digital vs Hybrid - On Chip Resources*

	Full Digital	Hybrid v1	Hybrid v2
Cells	105	5	4
Area	221	14	17

The advantage of a full digital controller is its design simplicity, a state machine with three states. However, in contrast to our Hybrid and Analog controllers, it is a significantly larger surface area. This is mainly due to needing two digital counters. Not only do these counters take up more surface area but they also leave our design susceptible to thermal fault attacks. Such attacks could be used to desynchronize our counters and power capacitor, resulting in unintended executions that could be used to leak information.

Our next iterative design was a hybrid controller, illustrated in Figure 4, that addressed the surface area and fault attacks of our digital controller. Table 1 shows the drastic difference in surface area between our digital and hybrid controllers. We observed that capacitors could be used as counters based on their natural electrical discharge leakage.[8] Replacing our digital counters with capacitors minimized our surface area and resolved our thermal fault-based attack concerns. Any thermal fault attacks would affect the timing and power capacitors in the same way. Allowing the capacitors to stay synchronized with respect to each other. In addition, if a desynchronization should ever happen both counters will continue to discharge and eventually trigger. Therefore the power capacitor will still be short and the timing window will be based on the counters, not the execution time.

The next design involved implementing a complete analog controller, illustrated in Figure 5. Unlike the hybrid controller, all the switching controllers were implemented in analog. As a result, two optimizations were implemented that are unavailable to a digital design. First, we used our counting capacitors to directly control are switching MOSFETs. Secondly, we used our GPIO to control our timing capacitor. When the processor initiates secure mode and the GPIO goes high, this would power and hold our recharge capacitor. Until the discharge capacitor triggers causing a short that would in turn brown out the processor. This would release the GPIO causing our recharge capacitor to begin to lose charge. Therefore the time

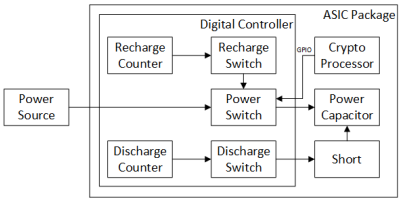

Figure 3: *Digital Block Diagram*

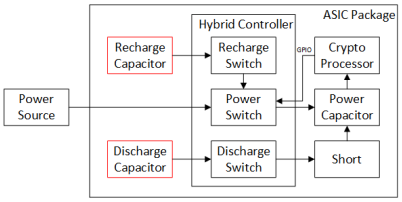

Figure 4: *Hybrid Block Diagram*

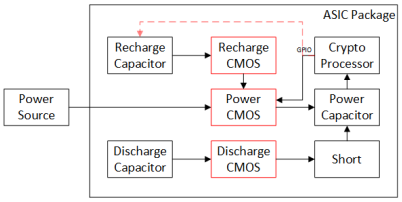

Figure 5: *Analog Block Diagram*

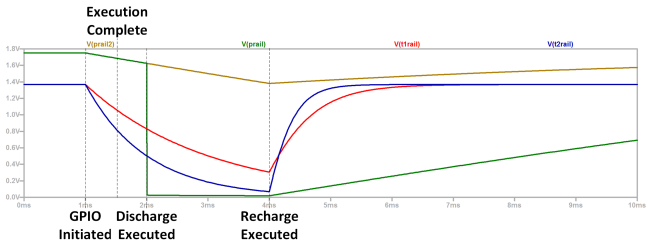

Figure 6: *SPICE simulation of the hybrid quantization controller.*

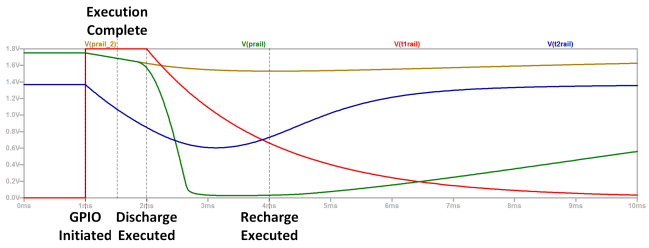

Figure 7: *SPICE simulation of the full-custom quantization controller.*

for our recharge cycle is not simply determined by the recharge capacitor but by a smaller recharge capacitor plus the discharge capacitor. Still secure timing per quantum but with a smaller capacitor. These design optimizations give the analog controller an advantage in terms of power and surface area but at the cost of performance. Since the capacitors are directly controlling the switching MOSFETs they do not switch instantly, but gradually as the capacitor continues to discharge. Each quantum cycle will have to be a little larger to accommodate the switching transition times, unlike a digital switch where the transitions states would be relatively instant. For this reason, we favor our hybrid controller design, which has a slightly higher surface area but much better performance.

5 EVALUATION

We evaluate our three quantization controller designs using a combination of Register Transfer Level (RTL) simulation and Simulation Program for Integrated Circuits Emphasis (SPICE) simulation. RTL simulation generates waveforms from our wholly-digital controller and the digital portion of our hybrid controller. SPICE simulation generates waveforms from our full-custom controller and the analog portion of our hybrid controller. To simulate the hybrid controller, we model the analog behavior as seen digitally for the RTL simulation and we model the digital control logic as switches in the SPICE simulation. For all simulations, we model the behavior of the GPIO from the processor.

5.1 SPICE Simulation

Figures 6 and 7 show the SPICE simulation of the hybrid and full-custom quantization controllers, respectively. These simulations demonstrate the three—execution independent—phases of operation of our quantization controllers:

(1) **Secure Mode**: GPIO triggered secure execution
(2) **Discharge Mode**: Counter 1 triggered power discharge
(3) **Recharge Mode**: Counter 2 triggered power recharge

The processor starts out executing in a continuously powered fashion, side-channels exposed. This mode, while insecure, allows the capacitors (power capacitor, discharge counter capacitor, and recharge counter capacitor) to fully-charge and eliminates software run time overhead. Then, software signals to the quantization controller that it wants to perform some secure operation by setting the GPIO high. When the GPIO goes high, our power switch disconnects the power capacitor form the chip's power rail—isolating the processor's power consumption from external package pins. Now disconnected, the power capacitor begins to discharge due to processor execution and leakage. Eventually, enough charge will dissipate from the power capacitor that the processor dies due to insufficient voltage. At some later time, dictated by the workload independent, natural leakage of the discharge counter capacitor, the discharge switch connects the power capacitor to our shorting circuit. The shorting circuit clamps the power capacitor's voltage to a fixed voltage (.5V) to prevent the amount of current required to recharge the power capacitor from revealing information about secure execution.[2] At an even later time, dictated by the workload independent, natural leakage of the recharge counter capacitor, the discharge switch opens and the power switch closes. This disconnects the power capacitor from the shorting circuit and reconnects it to the chip's power rail, recharging the capacitors and resuming full-speed, but insecure, execution. No matter the load used to simulate the energy requirement of secure execution, all events except the time the processor dies (which is not attacker observable), occur at the same time and all capacitors have a consistent charge when reconnected to the power rail.

As previously described in our design section, the full-custom implementation has two distinct design differences from the hybrid design. The analog design uses the GPIO line as both a control line and a capacitive counter that controls the shorting circuitry. Using our GPIO optimization the recharge counter does not begin to dissipate charge until the processor dies (i.e., stops driving the GPIO line). A drawback of this optimization is that chip designers need to carefully design the relative capacitance of the discharge and recharge capacitors to prevent counter inversion, i.e., recharge

[2] Had the capacitor not been shorted, it would have continued to discharge as indicated.

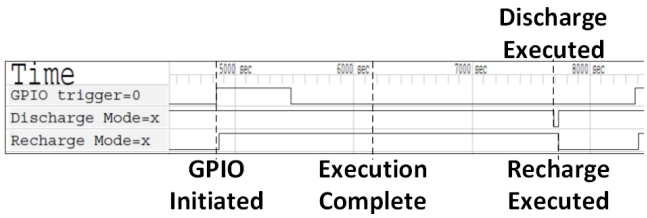

Figure 8: *RTL simulation of the wholly-digital and hybrid quantization controllers.*

occurring before discharge. Another drawback of directly using the capacitor counters to control the MOSFET switch is also illustrated by the slow discharge rate of our power capacitor. Though just like the hybrid simulation, the key observation is that between 1 ms and 4ms there is no connection between the main power rail and the internal power rail of the processor.

5.2 RTL Simulation

Figure 8 shows the resulting waveforms of the RTL simulation of our wholly-digital and hybrid quantization controllers. `GPIO trigger` is an input to the controller from the processor that informs the controller to enter secure execution mode. When `GPIO trigger` goes high and the controller is in the recharge mode, the controller opens the switch that connects the power capacitor to the chip's power rail. `Recharge Mode` is the output signal that drives this power isolation switch; when 1, the processor is isolated from the power rail, when 0, the power rail drives directly the processor and capacitors. After a workload independent amount of time executing in secure mode (ideally, some time after the secure execution completes), the quantization controller goes into discharge mode by closing the switch that connects the power capacitor to the shorting circuit. `Discharge Mode` is an output signal that when 0, connects the power capacitor to the shorting circuit and, when 1, disconnects the power capacitor to the shorting circuit. Once the power capacitor completely discharges, the cycle repeats.

Not shown in Figure 8 are the counters that control how long the controller stays in secure execution mode and how long it stays in discharge mode.[3] In the wholly-digital quantization controller, the counters are digital, while in the hybrid controller, the counters are the digital representation of the voltage across a capacitor that leaks charge naturally. While both implementations are workload independent, the capacitor-based counters have an advantage in that they encode a relative notion of time with respect to each other and the power capacitor—a critical property given temperature-based fault attacks.

6 FUTURE WORK

This paper provides the initial validation for a novel, isolation-based approach to addressing the most common side channels used to capture secrets from otherwise secure systems. Open questions remain about the effectiveness, practicality, and efficiency of our approach.

[3]There is no need for a counter to determine how long to stay in recharge mode because software controls that transition. This avoids adding unnecessary overhead when security is not required and reduces quantization controller complexity. A pull-down resistor ensures that `GPIO trigger` stays low when the processor is not powered.

To demonstrate the deployability of our approach, we are implementing it in hardware, as part of an existing cryptography processor circuit. This shows how circuit designers can incorporate it into their own designs. The central challenge is constructing the desired power and counter capacitors using CMOS. The most critical question that any side channel defense must answer is whether the defense is effective. Answering this question requires showing two things: that the defense eliminates the targeted side channels and quantifying the information increase in other side channels. We are addressing this question by showing that our approach eliminates power and timing side-channels in real systems. We are also developing a solution to eliminate the electromagnetic side channel that the shorting circuit creates. Lastly, we will quantify the runtime cost to software of forcing a decomposition into power and time quanta and using DINO [5].

7 CONCLUSION

This paper takes the first steps in showing how system designers can utilize the isolation and task-level atomicity guarantees provided by energy harvesting techniques to construct systems with well-defined power and side channel emanations that are robust against fault attacks. The experimental results show that, when using capacitors as counters, the hardware overhead of our approach is negligible, while gaining the benefit of resilience to thermal fault attacks. We see our isolation-based enabling the construction of more secure systems; whether they be ultra-low-power embedded devices or commodity servers.

ACKNOWLEDGMENTS

This material is based upon work supported by the Assistant Secretary of Defense for Research and Engineering under Air Force Contract No. FA8721-05-C-0002 and/or FA8702-15-D-0001. Any opinions, findings, conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the Assistant Secretary of Defense for Research and Engineering.

REFERENCES

[1] Alexei Colin and Brandon Lucia. 2016. Chain: tasks and channels for reliable intermittent programs. In *International Conference on Object-Oriented Programming, Systems, Languages, and Applications (OOPSLA)*. 514–530.
[2] Matthew Hicks. 2017. Clank: Architectural Support for Intermittent Computation. In *International Symposium on Computer Architecture (ISCA)*. 228–240.
[3] Paul Kocher, Joshua Jaffe, and Benjamin Jun. 1999. Differential power analysis. In *Annual International Cryptology Conference on Advances in Cryptology (CRYPTO)*. 789–789.
[4] Paul C Kocher. 1996. Timing attacks on implementations of Diffie-Hellman, RSA, DSS, and other systems. In *Annual International Cryptology Conference on Advances in Cryptology (CRYPTO)*. 104–113.
[5] Brandon Lucia and Benjamin Ransford. 2015. A Simpler, Safer Programming and Execution Model for Intermittent Systems. In *Conference on Programming Language Design and Implementation (PLDI)*. 575–585.
[6] Michael Moukarzel, Thomas Eisenbarth, and Berk Sunar. 2017. μLeech: a Side-Channel Evaluation Platform for IoT. In *International Midwest Symposium on Circuits and Systems*.
[7] Rambus. 2017. Licensed Countermeasures - Rambus. (2017). https://www.rambus.com/security/dpa-countermeasures/licensed-countermeasures [Accessed 28 July 2017].
[8] Kaiyuan Yang, Matthew Hicks, Qing Dong, Todd Austin, and Dennis Sylvester. 2016. A2: Analog malicious hardware. In *Symposium on Security and Privacy (Oakland)*. 18–37.