Mixed-Signal Hardware Security Using MixLock: Demonstration in an Audio Application

Julian Leonhard, Marie-Minerve Louërat, Hassan Aboushady, Ozgur Sinanoglu, Haralampos-G. Stratigopoulos

Mixed-Signal Hardware Security Using *MixLock*: Demonstration in an Audio Application

Julian Leonhard*, Marie-Minerve Louërat*, Hassan Aboushady*,
Ozgur Sinanoglu‡, Haralampos-G. Stratigopoulos*
*Sorbonne Université, CNRS, LIP6, Paris, France
‡New York University Abu Dhabi, Abu Dhabi, UAE

***Abstract*—In this paper, we demonstrate a recently proposed security technique for mixed-signal circuits in the context of a real application. The security technique, called *MixLock*, is based on logic locking of the digital section of the mixed-signal circuit and can be used as a countermeasure for reverse engineering and counterfeiting. We demonstrate *MixLock* in an audio application, where the underlying $\Sigma\Delta$ ADC that digitizes the audio input is locked. We show the effect that locking has on the recorded audio quality based on a metric that counts the resultant glitches per second and by also providing a link where the interested reader can download and listen to output audio samples for locked and unlocked versions of the $\Sigma\Delta$ ADC.**

I. Introduction

During its lifetime an integrated circuit (IC) may be subjected to various types of attacks. Fig. 1 shows the different stages of the lifetime of an IC and attacks that can be put into effect at each stage. Threats can be categorized into Hardware Trojans [1], reverse engineering [2], counterfeiting [3], and side-channel attacks [4], [5]. Hardware Trojans refer to malicious hardware inserted into the design to leak secret information, degrade performance or lead to complete malfunction, i.e., denial of service. Reverse engineering consists in extracting the netlist and other technological information of the IC. It is implemented via a series of steps that include de-packaging, de-layering, imaging, aligning and stitching the images of the different layers, and software tools for final netlist extraction. Reverse engineering aims at reducing the attacker's technological disadvantage compared to its competitors, producing a cloned counterfeit, or understanding the design to mount an attack for stealing secret information. Counterfeiting includes cloning chips and selling them as original, a foundry overproducing and illegitimately selling chips beyond the number agreed on in the contract with the IC design house, a test facility remarking failing chips that should be discarded and illegitimately selling them with forge documentation, and recycling used chips that are possibly aged with degraded performance and reselling them as new. Side-channel attacks aim at inferring cipher keys and other sensitive data and secret information or injecting faults in the computation in order to degrade performance or lead to denial of service.

The attacks can have serious implications on governments (i.e., national security threat if attacked chips are deployed in sensitive sectors, such as defense and infrastructure), on industry (i.e., loss of revenue due to loss of know-how and intrusion of counterfeits in the market), and on the society and consumers (i.e., low-quality counterfeited products).

Hardware security aims at understanding security breaches in ICs and developing mechanisms for detecting attacks or preventing them by implementing countermeasures for on-chip resilience. Hardware security has been studied extensively for digital ICs recently [1]–[5], but for analog, mixed-signal, and RF ICs, only a few methodologies have been proposed; the solution space is still largely unexplored [6], [7].

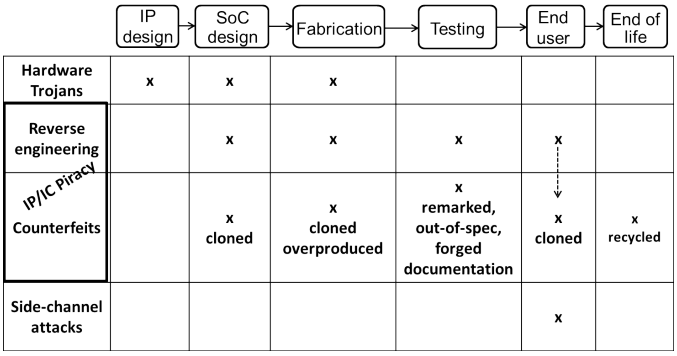

Fig. 1. Hardware attacks at different stages of the lifetime of an IC.

In this work, we focus on the problem of IC/IP piracy, which includes reverse engineering and counterfeiting. In particular, we develop a countermeasure for mixed-signal IC/IP piracy that is based on design locking. Locking aims at modifying the design, in order to introduce k key bits. There is only one valid combination of key bits, i.e. the secret key, that can result in correct functionality for any input. Otherwise, if an invalid key is applied, then the functionality will be corrupted for some or all inputs.

There are various techniques for locking digital ICs, known as logic locking techniques. The earliest logic locking techniques aimed at inserting key-gates into the design [8], i.e. XOR and XNOR gates, controlled by the key bits, as shown in Fig. 2. Researchers are working in parallel trying to show the vulnerabilities of existing logic locking techniques, proposing attacks that can break them: (i) The brute-force attack sequentially applies keys until the valid one is found. This attack can be circumvented by using a large k (typically at least 64 bits); (ii) The SAT attack, the most lethal attack based on a Boolean satisfiability solver, can recover the secret key with very reasonable effort; (ii) Removal attacks aim at identifying and removing the added protection logic, i.e., the gates K_1-K_3 in Fig. 2; (iv) Approximate attacks aim at extracting a key that establishes an incorrect yet approximate functionality. The most recent state-of-the-art logic locking technique is Stripped-Functionality Logic Locking (SFLL) [9] and provides quantifiable resilience against these attacks.

The secret key management scheme is common for all logic locking techniques and includes storing the secret key on-chip in a tamper-proof memory or generating it on-chip; in the

978-1-7281-1201-5/19/$31.00

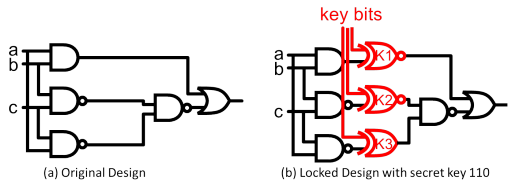

Fig. 2. Random logic locking.

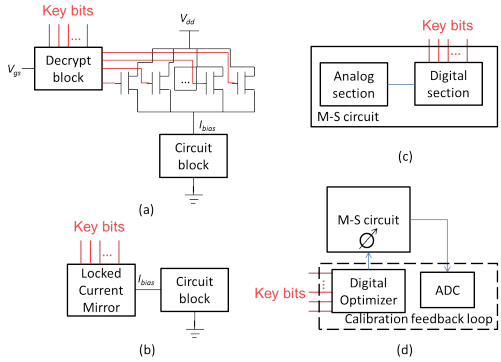

Fig. 3. Locking techniques for analog ICs: (a) obfuscating biasing transistors [11]; (b) locking current mirrors [12]; (c) logic locking of the digital section of a mixed-signal IC [13]; (d) logic locking of the digital optimizer in the calibration feedback loop [14].

latter case, a Physical Unclonable Function (PUF) [10] can be utilized to even produce chip-unique keys. Chip activation can be done by the trusted design house, which arguably is a practical solution only for low-volume fabrication. For high-volume fabrication, activation needs to be done on the test floor concurrently during automated testing. For protecting the key against an untrusted test facility, the design house can remotely activate the chips using asymmetric cryptography [8].

Locking analog ICs is very challenging as the key bits need to be introduced in a way that nominal performance is not degraded. To date, four approaches have been proposed, as shown in Fig. 3. More specifically, in [11], it is proposed to replace biasing transistors with parallel-connected transistors whose gates are controlled by key bits. The key bits enable transistors whose aggregate width equals that of the original biasing transistor. In [12], extra branches are inserted into the current mirror, where each branch is comprised of the mirroring transistor and possibly several switches that are controlled by the key bits. The resultant biasing circuit will depend on which branches are switched-on, as well as on the geometry of the mirroring transistor in these branches. In [13], the *MixLock* technique is proposed that locks the functionality of a mixed-signal circuit via logic locking of its digital section. In [14], it is proposed to lock the digital optimizer in the calibration feedback loop such that it provides the wrong tuning settings. All these locking approaches are vulnerable to removal attacks since a smart attacker can simply remove the locked blocks and replace them with "fresh" ones with no locking mechanism. Perhaps [13] provides the highest resilience against removal attacks since the design of the digital section is intertwined with the design of the analog section, thus replacement by redesign is not straightforward. In addition, the locking approaches in [11], [12] act on the biasing of the circuit and, thereby, the attacker does not have to recover the key; it suffices to recover the biases, which are typically small in number.

The objective of this paper is to demonstrate *MixLock* [13] in a real application context. In particular, we built a demonstrator that emulates a microphone for capturing a sound source, signal processing for digitizing the input audio, and a speaker for listening back the sound source. *MixLock* is used to lock the ADC in the signal processing chain. The effect of locking on audio quality can be measured by the glitches introduced from the locking operation. The interested reader can also download audio samples to listen to the effect of locking.

The rest of the paper is structured as follows. In Section II, we provide a brief overview of *MixLock*. In Section III, we describe the demonstrator. In Section IV, we demonstrate the impact of locking on audio quality. Finally, Section V concludes the paper.

II. MixLock with SFLL

MixLock aims at securing mixed-signal circuits through a logic locking mechanism applied to the circuit's digital section as illustrated in Fig. 3(c). Only when the valid key is provided the mixed-signal circuit performs its intended function. Otherwise, for invalid keys the mixed-signal performances are pushed outside of their specification, i.e., they are locked.

The term *analog security* is used to quantify the efficiency of mixed-signal performance locking, while the term *digital security* is used to quantify the resilience against logic locking attacks in the digital domain.

MixLock presents several appealing properties. It is non-intrusive since it does not alter the analog section, which is key for its wide adoption by designers. Modifications in the digital section do not affect mixed-signal performance either. It incurs low area and power overheads since area and power are dominated by the analog section which is left intact. It is fully-automated since logic locking adds only one extra synthesis step. Finally, this concept is applicable to a wide range of mixed-signal circuits such as PLLs, RF transceivers, data converters, etc.

Mixlock provides protection against reverse engineering and certain types of counterfeiting, namely cloning, overproducing, and remarking. It can provide protection against recycling if the key management scheme supports chip-unique keys that should be loaded every time at start-up.

Breaking *Mixlock* will require either recovering the secret key via a logic locking attack or trying to unlock directly mixed-signal performances by applying an iterative multi-objective optimization algorithm. The latter is unlikely to succeed since mixed-signal performances do not show a smooth monotonic relationship with the key bits. Regarding the former attack, *Mixlock* is independent of the underlying logic locking technique, but to achieve strong digital security *Mixlock* uses the state-of-the-art SFLL logic locking mechanism [9].

The architecture of SFLL is shown in Fig. 4 for a digital circuit with n inputs. A checker is introduced into the design that flips the output of the original circuit for all input patterns that are a Hamming distance (HD) h away from the secret key of k-bits. The secret key is hard-coded and after synthesis the checker is immersed in the original circuit and, thereby, the

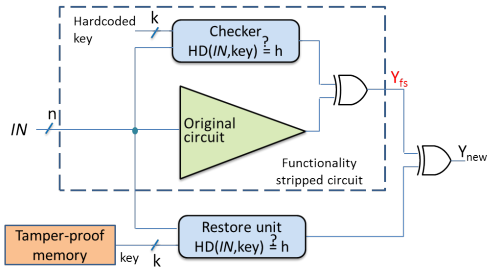

Fig. 4. SFLL architecture for digital circuits.

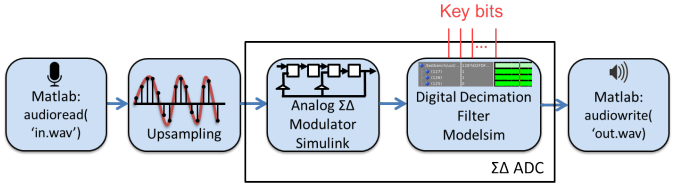

Fig. 5. *Mixlock* demonstration in an audio application.

two become inseparable. A restore unit is used that also checks the HD between the input and the secret key, but this time the secret key is provided directly at the input of the restore unit. For example, a possible key management scheme, as explained in the introduction, is to store the secret key in a tamper-proof memory, as shown in Fig. 4. The restore unit corrects the functionality only with the correct secret key loaded in the tamper-proof memory.

In terms of resiliency against logic locking attacks, it can be shown that the SAT attack effort required to extract the secret key is equivalent to breaking a $k - \lceil \log_2 \binom{k}{h} \rceil$-bit key in a brute-force way. The resiliency against the removal attacks is proportional to the number of protected input patterns, i.e. the input patterns that produce an erroneous output for an invalid key; the larger the number of protected input patterns, the more intricate the changes to the original logic are, and, thereby, the harder it is for the removal attack to succeed. It can be shown that the number of protected input patterns is $\binom{k}{h} \cdot 2^{n-k}$. Finally, the resiliency against approximate attacks is proportional to the error rate or functionality corruption at the output; the higher the error rate, the more difficult it is to find a key that establishes approximate functionality. It can be shown that the error rate is $\frac{\binom{k}{h}}{2^k}$. As it can be seen from the resilience expressions above, SFLL allows a designer to trade-off the desired digital security level against different attacks by choosing appropriately k and h.

III. *Mixlock* Demonstration in an Audio Application

In [13], *Mixlock* was demonstrated on a band-pass (BP) ΣΔ ADC case study. A ΣΔ ADC is decomposed into a ΣΔ modulator, which is the analog section, and a decimation filter, which is the digital section. In this case, *Mixlock* naturally locks the decimation filter. It was shown that *Mixlock* achieves strong analog security based on one thousand randomly chosen invalid keys; the main Signal-to-Noise Ratio (SNR) performance was degraded dramatically below its specification. The decimation filter was locked so as to guarantee, in addition, strong digital security.

In this paper, we demonstrate *Mixlock* in an audio application, in order to evaluate the impact of locking on the audio quality. This demonstrator helps us in essence to *listen to* the effect of locking. The demonstrator is illustrated in Fig. 5. An audio sample is read from the microphone of the PC and thereafter it is captured and sampled using the Matlab function `audioread`. Upsampling based on linear interpolation is used so as to artificially smoothen the signal so as to be able to present it to a ΣΔ ADC for a second digitization. The output of the ΣΔ ADC can be heard directly from the speaker of the PC using the Matlab function `audiowrite`. The ΣΔ modulator in this demonstrator is a second-order low-pass (LP) continuous-time ΣΔ modulator. The decimation filter is the same one used in [13]. The ΣΔ modulator is modelled in Simulink, while the decimation filter is modeled with VHDL and is simulated in Modelsim.

For locking the decimation filter we use the exact same settings as in [13]. More specifically, SFLL essentially locks a single bit line stripping the functionality of the sub-circuit that drives the bit line. For the decimation filter, we experiment with two different locking approaches applied in the first block in the digital signal processing chain, which is a comb filter. The rationale is that in this way errors will be propagated and spread out to the rest of the circuit, introducing high functionality corruption. In the first approach, we use a single SFLL mechanism with $k = 128$ and $h = 15$ that locks the most significant (MSB) bit of the comb filter's output. This approach provides a 64-bit resilience against the SAT attack and sufficient resilience against removal and approximate attacks, as dictated by the formulas in Section II. In the second approach, which is called 1.5xSFLL, this first SFLL mechanism is combined with a second SFLL mechanism with $k = 32$ and $h = 16$ that locks the MSB-1 bit of the comb filter's output. 1.5xSFLL increases significantly the number of protected patterns and the error rate, i.e., it increases functionality corruption. In theory, however, 1.5xSFLL can be reduced to the single SFLL mechanism [13], so it is appropriate only for the naive attacker. Locking the decimation filter with a single SFLL mechanism results in an area and power overhead of 6.7% and 9.8%, respectively, while 1.5xSFLL results in a slightly higher overhead. No SNR degradation is observed compared to a design with no locking mechanism.

In this demonstrator, the SNR metric cannot be used for quantifying analog security as was shown in [13]. The reason is that SNR requires a sinusoidal input, while audio signals are time-varying in nature; their spectral contents vary with time, they are rich in frequencies, etc. For this purpose, we use an audio quality metric to evaluate the effect of locking, namely glitches per second (GPS). In essence, unless the valid secret key is applied to the decimation filter, locking introduces errors that get translated into audible glitches in the output signal.

IV. Results

Our experiment involves processing audio samples through the system in Fig. 5 and examining the effect of locking on the audio quality. Table I shows in the first two columns the different audio samples that were employed and their duration. Audio samples include speech recordings in German and English and professional music recordings of various genres. The third and fourth columns show the GPS for a locked ΣΔ ADC using the two locking approaches discussed in

TABLE I
EFFECT OF LOCKING BY MIXLOCK ON AUDIO QUALITY

Audio sample	Duration [s]	SFLL GPS [1/s]	1.5xSFLL GPS [1/s]
German Voice (counting)	13	0.0	7046
English Voice (*MixLock* goal)	4	0.0	7309
Bob Marley - No Woman No Cry	15	0.3	7511
Benny Goodman - Bugle Call Rag	15	0.3	7529
Kenny Ball - I Wanna Be Like You	15	3.1	7559
John Coltrane - Nature Boy	15	0.9	7543
Beethoven - Symphony No. 9	15	0.4	7539

Section III, namely SFLL and 1.5xSFLL. The interested reader can also download and listen to the output audio samples from this link: https://nuage.lip6.fr/s/CYowe89aXBe6rsP. The downloadable archive includes the output audio samples in the case of the unlocked design, where the valid key is applied, and two locked designs using SFLL and 1.5xSFLL, where a random invalid key is applied.

As it can be seen, 1.5xSFLL corrupts dramatically the audio quality resulting in very frequent glitches. In fact, the recording gets buried under the noise level and is hardly recognizable. SFLL results in noticeable glitches for the music recordings that can be heard as noisy "cracks". However, for the voice recordings, no glitches are noticed.

This result can be explained as follows. By default SFLL corrupts the output of the targeted digital circuit for some and not all input patterns. In our case, we have $n = k = 128$ and $h = 15$, thus the number of protected patterns is $\binom{k}{h} \cdot 2^{n-k} \approx 1.32 \cdot 10^{19}$, which is a very small subset of all possible $2^{128} \approx 3.4 \cdot 10^{38}$ input combinations. An analog input, i.e., an audio signal in our case, gets translated into a sequence of patterns at the input of the protected digital block within the decimation filter. Since music recordings have higher signal activity compared to voice recordings, it turns out that they get translated to a larger number of distinct patterns at the input of the protected digital block. Thus, the probability of hitting protected input patterns is higher, resulting in a higher probability of audio quality corruption. Note that short duration samples were recorded for practical purposes and that for longer duration samples SFLL is expected to also result in glitches when voice is processed.

At this point it is important to recall the purpose of hardware locking and hardware obfuscation in general. The aim is not necessarily to encrypt the data that is processed by the hardware, i.e., corrupt audio quality to bare noise. The aim is to render the hardware low-quality and unusable unless the valid secret key is known; i.e., glitches occurring at regular and frequent intervals are sufficient.

Finally, Fig. 6 shows an excerpt of the transient waveform of the output audio in the case of the English voice recording stimulus. Two waveforms are plotted for the unlocked design and the design locked with 1.5xSFLL. The large number of glitches introduced by 1.5xSFLL can be easily identified.

V. CONCLUSION

In this paper we demonstrated the effect of locking a mixed-signal circuit that is part of the signal processing chain in an audio application. The recently proposed *MixLock* locking technique for mixed-signal circuits was used. The effect of locking is measured by the number of occurring glitches, but it can be also clearly heard in audio samples that are provided. We demonstrate that locking results in disturbing glitches, rendering the device low-quality and unusable unless the valid secret key that unlocks the design is known. To the best of our knowledge, this the first demonstrator showing the effect of locking on a circuit in a way that can be perceived by humans via a sense.

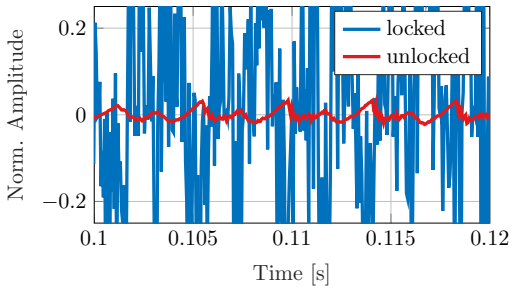

Fig. 6. Frequent glitches introduced by 1.5xSFLL in the transient waveform of an output audio sample.

ACKNOWLEDGMENTS

This work has been carried out in the framework of the ANR STEALTH project with N° ANR-17-CE24-0022-01. J. Leonhard has a fellowship from the doctoral school EDITE de Paris.

REFERENCES

[1] S. Bhunia et al., "Hardware Trojan attacks: Threat Analysis and Countermeasures," *Proceedings of the IEEE*, vol. 102, no. 8, pp. 1229–1247, 2014.
[2] R. Torrance and D. James, "The State-of-the-Art in Semiconductor Reverse Engineering," in *IEEE/ACM Design Automation Conference*, 2011, pp. 333–338.
[3] U. Guin et al., "Counterfeit integrated circuits: A rising threat in the global semiconductor supply chain," *Proceedings of the IEEE*, vol. 102, no. 8, pp. 1207–1228, 2014.
[4] K. Tiri and I. Verbauwhede, "A VLSI Design Flow for Secure Side-Channel Attack Resistant ICs," in *Design, Automation & Test in Europe*, 2005, pp. 58–63.
[5] A. Barenghi et al., "Fault Injection Attacks on Cryptographic Devices: Theory, Practice, and Countermeasures," *Proceedings of the IEEE*, vol. 100, no. 11, pp. 3056–3076, 2012.
[6] A. Antonopoulos et al., "Security and trust in the analog/mixed-signal/RF domain: A survey and a perspective," in *IEEE European Test Symposium*, 2017.
[7] M. M. Alam et al., "Challenges and Opportunities in Analog and Mixed Signal (AMS) Integrated Circuit (IC) Security," *Journal of Hardware and Systems Security*, vol. 2, no. 1, pp. 15–32, 2018.
[8] J.A. Roy et al., "Ending Piracy of Integrated Circuits," *IEEE Computer*, vol. 43, no. 10, pp. 30–38, 2010.
[9] M. Yasin et al., "Provably-secure logic locking: From theory to practice," in *ACM/SIGSAC Conference on Computer & Communications Security*, 2017, pp. 1601–1618.
[10] C. Herder et al., "Physical unclonable functions and applications: A tutorial," *Proceedings of the IEEE*, vol. 102, no. 8, pp. 1126–1141, 2014.
[11] V. V. Rao and I. Savidis, "Protecting analog circuits with parameter biasing obfuscation," in *IEEE Latin American Test Symposium*, 2017.
[12] J. Wang et al., "Thwarting analog IC piracy via combinational locking," in *IEEE International Test Conference*, 2017.
[13] J. Leonhard et al., "Mixlock: Securing mixed-signal circuits via logic locking," in *Design, Automation & Test in Europe Conference*, 2019.
[14] N. G. Jayasankaran et al., "Towards provably-secure analog and mixed-signal locking against overproduction," in *IEEE/ACM International Conference on Computer-Aided Design*, 2018.