Digitally-Assisted Mixed-Signal Circuit Security

Julian Leonhard, Nimisha Limaye, Shadi Turk, Alhassan Sayed, Alán Rodrigo Díaz-Rizo, Hassan Aboushady, Ozgur Sinanoglu, Haralampos-G. Stratigopoulos

Digitally-Assisted Mixed-Signal Circuit Security

Julian Leonhard, Nimisha Limaye, *Graduate Student Member, IEEE*, Shadi Turk, Alhassan Sayed, Alán Rodrigo Díaz Rizo, Hassan Aboushady, *Senior Member, IEEE*, Ozgur Sinanoglu, *Senior Member, IEEE*, and Haralampos-G. Stratigopoulos, *Member, IEEE*

***Abstract*—The design and manufacturing steps of a chip typically involve several parties. For example, a chip may comprise several third-party Intellectual Property (IP) cores and the Integrated Circuit (IC) fabrication may be outsourced to a third-party foundry. IP cores and ICs are shared with potentially untrusted third parties and, as a result, are subject to piracy attacks. Even more, any legally purchased chip may be reverse-engineered to retrieve the design down to transistor-level and, thereby, it is also subject to piracy attacks. In this paper, we propose *MixLock*, an anti-piracy countermeasure for mixed-signal IP cores and ICs. *MixLock* protection is based on inserting a lock mechanism into the design such that correct functionality is established only after applying a key which is the designer's secret. The lock mechanism acts on the mixed-signal performances by leveraging logic locking of the digital part. *MixLock* presents several key attributes. It is generally applicable, it is non-intrusive to the sensitive analog section, it incurs no performance penalty and has very low area and power overheads, it is fully automated, and it is capable of co-optimizing security in both the analog and digital domains. We demonstrate *MixLock* on a $\Sigma\Delta$ Analog-to-Digital Converter (ADC) using hardware measurements and an audio demonstrator.**

***Index Terms*—Hardware security and trust, mixed-signal integrated circuits, IP/IC piracy, locking.**

I. Introduction

The globalization of the semiconductor industry where many design, manufacturing, and test tasks are outsourced to third-parties, as well as the increasing capabilities for reverse-engineering a chip [1], have given rise to several IP/IC piracy threat scenarios [2]. For example, a System-on-Chip (SoC) integrator that receives the blueprint of an IP may clone the IP, i.e., produce a similar or identical version of the original IP, and illegitimately reuse it in another SoC without remunerating the IP provider. A foundry that also receives the blueprint of a SoC and all its sub-IPs can also perform the cloning operation. In addition, it can overproduce chips beyond the number agreed on in the contract and thereafter sell them illegally. A test facility may remark out-of-spec chips as functional, possibly with a forged datasheet, and sneak them illegally into the market. An end-user may reverse-engineer a legally purchased chip to extract the required proprietary information, i.e., architecture, netlist, and layout, needed to perform the cloning operation. Finally, a scrapped chip, which is likely to have aged and show degraded performance, can be recycled and can re-enter the market as a seemingly fresh and unused chip. IP/IC piracy includes all these types of counterfeiting, i.e., cloning, overproducing, remarking, and recycling. IP/IC piracy is a major preoccupation nowadays for the industry (e.g., lost sales, know-how, and brand value), governments (e.g., national security threat if counterfeits are used in critical infrastructure), and the society as a whole (e.g., safety concerns due to low-quality counterfeits).

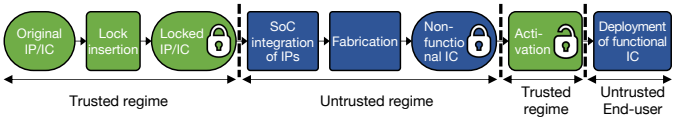


Fig. 1: Standard locking flow.

In the last decade, extensive research has been carried out to understand trust and security threat scenarios in digital ICs. Analog and mixed-signal (A/M-S) ICs on the other hand have so far received far less attention and as a consequence the number of proposed solutions is low [3]. This work proposes *MixLock*, a method of hardware security for M-S ICs based on locking with the goal to prevent IP/IC piracy.

Locking is an end-to-end protection against potential attackers located anywhere in the IC supply chain [4]. A standard locking flow is depicted in Fig. 1. The IP/IC owner transforms the circuit by embedding into it a lock mechanism that receives as input a key in the form of a bit-string. The circuit's functionality becomes a function of the key. The correct key that restores the nominal functionality is the designer's secret. A potentially untrusted SoC integrator or foundry receives the locked circuit and without knowing the secret key possesses a non-functional circuit, thus thwarting any cloning or overproduction possibilities. Thereafter, the IP/IC owner activates the fabricated chip by storing the secret key in a Tamper-Proof Memory (TPM). An end-user who reverse-engineers the chip cannot read the secret key from the TPM, thus possesses a non-functional circuit.

Manuscript received March 19, 2021; revised June 4, 2021, and July 16, 2021; accepted August 24, 2021. This work was supported by the ANR STEALTH project under Grant ANR-17-CE24-0022-01. The work of J. Leonhard was supported by the Doctoral School EDITE de Paris through Fellowship. The work of A. R. Díaz Rizo was supported by the Mexican National Council for Science and Technology (CONACYT) through Fellowship. This article was recommended by Associate Editor S. Ghosh. (Corresponding author: Haralampos-G. Stratigopoulos.)

Julian Leonhard, Alán Rodrigo Díaz Rizo, Hassan Aboushady, and Haralampos-G. Stratigopoulos are with the Sorbonne Université, CNRS, LIP6, 75005 Paris, France (e-mail: julian.leonhard@lip6.fr; Alan-Rodrigo.Diaz-Rizo@lip6.fr; hassan.aboushady@lip6.fr; haralampos.stratigopoulos@lip6.fr).

Nimisha Limaye is with the Department of Electrical and Computer Engineering, Tandon School of Engineering, New York University, Brooklyn, NY 11201 USA (e-mail: nsl278@nyu.edu).

Shadi Turk is with Seamless Waves, 75020 Paris, France (e-mail: shadi.turk@seamlesswaves.com).

Alhassan Sayed is with the Sorbonne Université, CNRS, LIP6, 75005 Paris, France, and also with the Electronics and Communications Department, Minia University, Minia 61519, Egypt (e-mail: alhassan.sayed@lip6.fr).

Ozgur Sinanoglu is with the Division of Engineering, New York University Abu Dhabi, Abu Dhabi, UAE (e-mail: ozgursin@nyu.edu).

Digital Object Identifier 10.1109/TCAD.2021.XXXXXXX

MixLock leverages proven logic locking methods from the digital domain. In particular, we propose locking a M-S IC via logic locking of its digital part, while leaving the analog section intact. While *MixLock* is agnostic to the particular logic locking technique used, we use a state-of-the-art technique, namely Dishonest Oracle (DisORC) in combination with Truly Random Logic Locking (TRLL) [5]. We provide metrics for assessing the security in the analog domain, such as the impact of locking on the M-S functionality or the percentage of failing incorrect keys. The digital security level is expressed in terms of the resilience against known logic locking attacks. We showcase the technique in a hardware experiment using a $\Sigma\Delta$ ADC as a case study, and in an audio demonstrator where one can *listen* to the effect of locking the ADC in the audio signal processing chain. Overall, *MixLock* presents a number of attractive features, such as wide applicability, non-intrusiveness to the analog section, unaltered analog design flow, full automation, no performance penalty, and justifiable power and area overheads.

MixLock was originally proposed in [6], where the Stripped-Functionality Logic Locking (SFLL) technique [7] was employed for locking the digital section. We refer to this first implementation of *MixLock* as *MixLock 1.0*, while we refer to the implementation in this paper that employs DisORC in combination with TRLL as *MixLock 2.0*. Throughout this paper we refer to *MixLock*'s generic and locking mechanism-agnostic methodology simply as *MixLock*.

With respect to [6], this paper provides the following advances:

- The choice of logic locking plays a significant role in the context of *MixLock*. In the *MixLock 1.0* implementation, SFLL was carefully tuned to achieve high security in both digital and analog domains. Although invalid keys resulted in M-S functionality corruption, that is, specifications were violated, the M-S performance degradation was very moderate. For example, this is made evident when listening to the effect of locking in the audio demonstrator. The "locked" sound sample is still recognisable containing only some infrequent glitches. As remedy, a second SFLL mechanism, tailored towards increasing functionality corruption, was instantiated, although this second mechanism can be broken with reasonable effort. The difference between applying logic locking to a purely digital IC and applying logic locking in the context of *MixLock* is that in the former case a single bit-flip due to the use of an incorrect key is enough to cause an application crash, while in the latter case many computational errors need to be introduced into the digital section by the locking mechanism to be perceivable in an analog waveform. We will explain in detail why SFLL applied to *MixLock* falls short in this regard. To this end, using DisORC with TRLL in the *MixLock 2.0* implementation helps us to effortlessly meet this objective. DisORC with TRLL results in maximum digital security, while the M-S performance plummets for invalid keys.
- The ADC case study is extended from simulation in [6] to a full hardware experiment in this paper.
- We estimate an upper bound on the number of user keys that can establish correct functionality based on their Hamming Distance (HD) from the valid secret key. This is possible in the *MixLock 2.0* implementation as functionality corruption is a function of the user key while being independent of the input data. In contrast, in the *MixLock 1.0* implementation, functionality corruption was independent of the user provided key while being highly dependent on the input data, thus not allowing a bound on valid user keys to be estimated.
- We provide a comparison of the different A/M-S IC locking approaches in terms of generality, design complexity, overheads, and attack resiliency.

The rest of the paper is organized as follows. In Section II, we discuss the prior art on A/M-S IC locking. In Section III, we present the proposed *MixLock* technique. In Section IV, we present the hardware experiment. In Section V, we present the audio demonstrator. In Section VI, we provide a comparison of existing A/M-S IC locking techniques. Section VII concludes the paper.

II. State-of-the-Art in A/M-S IC Locking

A. Locking defenses

1) Biasing locking: The majority of analog locking techniques consider the insertion of a lock into the biasing circuitry. Providing the correct biases to the analog core is essential, given that the core will only function as foreseen if operated in the regime, i.e., the DC operating point, it was originally designed for.

Two types of approaches can be found in the literature, namely expanding the biasing circuit [8]–[10], and the design of a standalone block that generates the desired bias [11], [12].

In [8], a current source transistor is replaced with an array of parallel-connected transistors. The number of conducting transistors is controlled by a key. Thereby, in the case of a correct key, the aggregate width of the active transistors equals that of the replaced transistor, whereas a wrong key can lead to a different aggregate width, generating an incorrect bias current.

In [9], it is shown how to redesign a current mirror to embed the lock mechanism. The mirror transistor is replaced with a structure composed of several branches. Each branch contains a mirror transistor with an individual current mirroring ratio and several pass transistors controlled by the key-bits. The resultant bias current depends on which branches are "on" and the geometry of the mirror transistors of the "on" branches.

In [10], extending [8], a transistor is replaced with a number of parallel and series-connected transistors, thereby creating a mesh. Each transistor in the mesh is sized differently and is controlled with a key-bit. The aggregate width and length of such a mesh of transistors is then a function of the key.

In [11], it is proposed to secure a sense amplifier by using memristor-based circuitry to program the matched transistor pair's body-biasing voltage, used to eliminate the offset of the amplifier arising due to process variations. A memristor crossbar is programmed via a secret key allowing in turn the programming of a voltage divider that generates the body biasing. When applying an incorrect key the breakdown voltage of the emerging technology device can be reached, thus limiting the number of trials an attacker has for an attack.

In [12], an on-chip neural network is used to generate the bias. The neural network receives as input an analog key in the form of DC voltages and is trained to implement an impulse function at the correct key, i.e., only when the correct DC voltages are provided at the key inputs will the neural network generate the correct biases at its output, while otherwise the output remains constantly at an incorrect level.

2) Calibration locking: Calibration locking mechanisms act on the programming or configuration of the circuit, thus blocking the compensation against process variations and the correct setting of the desired operation mode.

In [13], it is proposed to apply logic locking in the digital optimizer block which is part of a calibration feedback loop. The calibration feedback loop comprises sensors to extract measurements, ADCs to digitize the measurements, digitally-controlled tuning knobs, and the digital optimizer that maps the measurements to selected tuning knobs so as to reduce the impact of process variations on the performances. An incorrect key will result in a non-calibrated circuit showing performance degradation.

In [14], it is proposed to exploit the naturally available programmability fabric of a highly-digitized A/M-S IC in order to lock it. It is argued that inserting a lock is not strictly necessary, since the configuration settings can serve as secret key-bits. Furthermore, the calibration algorithm to determine the circuit's correct configuration is kept secret and must be sufficiently complex so it cannot be reverse engineered.

In [15], Analog Floating-Gate Transistors (AFGTs) that serve to calibrate the circuit are used for inserting a lock. When starting the unlocking procedure, the tuning range of the AFGTs is reduced and their full tuning range is only unlocked when a number of secret waypoints is followed. To follow these waypoints, the AFGTs must be programmed in a certain order and with certain voltage levels. The waypoints, i.e., the secret programming order and AFGT voltage levels, therefore constitute a secret key in the analog domain. Only when all waypoints have been tracked, the complete programming range of the AFGTs is unlocked.

3) Mixed-signal locking: It aims at locking M-S ICs via logic locking of their digital section. Logic locking of the digital optimizer in the calibration feedback loop [13] and *MixLock* [6] fall into this category. The analog performance breaks unless the digital blocks are unlocked with the valid key. While biasing and calibration locking methods act on periphery circuits, i.e., biasing circuit and calibration mechanism, *MixLock* inserts the lock inside the core of the circuit, thereby acting on the signal processing. *MixLock* will be described in detail in Section III.

4) Compound locking: In [16], a compound locking method is proposed where a M-S circuit is locked via logic locking of its digital part and biasing locking of its analog part. Making analog and digital sections share an identical key, may stop an attacker from breaking the analog and digital blocks' locking mechanism independently.

B. Counter-attacks on locking

In parallel to developing A/M-S IC locking defenses, the first counter-attacks have surfaced recently.

The majority of them target biasing locking [17]–[20]. The counter-attack in [17] is demonstrated to break all existing biasing locking techniques. To recover the secret key, the attacker needs to derive three circuit equations that are solved thereafter using a Satisfiability Modulo Theory (SMT) solver. These equations include the relationships between: (a) the obfuscated component value, e.g., the geometry of the mirror transistor in a current mirror, and the key-bits; (b) the bias and the obfuscated component value; and (c) the bias and the performances listed in the data-sheet. Two counter-attacks are proposed in [17] that apply to the biasing locking techniques in [8]–[10]. The first exploits the key spacing, in particular the fact that there is an exclusion zone around the obfuscated parameter value, such that only the correct key produces values within this zone. The second exploits the monotonic relationship between a performance and the obfuscated component value when a single component is being obfuscated. Recommendations are also given on how to mitigate these two counter-attacks. The counter-attacks in [19], [20] are the most powerful since they are generally applicable to any biasing locking scheme, they cannot be mitigated, and also, compared to [17] where the attacker must derive circuit equations, are easier to execute even by a weak attacker with no particular circuit design expertise. The underlying idea is to remove the locked biasing circuit and replace it with an unlocked version of it and, thereafter, use optimization to synthesize the biasing circuit together with the circuit core. Both attacks use Genetic Algorithms (GAs) to perform the optimization. They differ in the way the fitness function is defined. In [19], the optimization aims at matching the frequency or transient response to that of the oracle, whereas in [20] the optimization aims at satisfying the performance trade-off promised in the datasheet without requiring an oracle. In [19], a second optimization is performed aiming at recovering the secret key by using the extracted obfuscated component value as a second fitness criterion. In this way, the search in the large space of keys, i.e., 2^n where n is the key size, can converge in reasonable time.

In [17], an attack on the existing mixed-signal locking techniques in [6], [13] is also proposed. Both these techniques use SFLL [7] as their underlying logic locking technique. The attack exploits the fact that SFLL corrupts the output of the digital section only for a handful of input patterns. The attacker finds this set of protected input patterns (PIPs) by analyzing the analog–digital interface signals and then determines the secret key using Boolean satisfiability (SAT) formulations. The attack is demonstrated for the calibration locking technique in [13] and it is claimed that it works also for the *MixLock 1.0* implementation in [6].

C. Physical obfuscation

Besides locking methods, other anti-piracy methods include split manufacturing [21] and camouflaging [22], [23]. While locking is an end-to-end protection method against all potential threats, i.e., untrusted SoC integrator, foundry, and end-user, split manufacturing protects only against an untrusted foundry and camouflaging protects only against an untrusted end-user that attempts to clone the circuit via reverse-engineering.

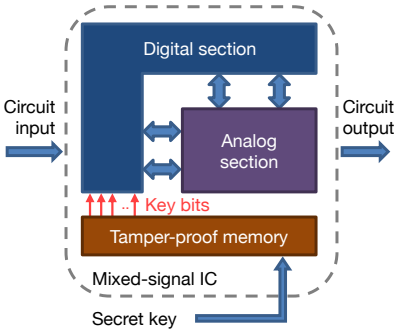


Fig. 2: *MixLock* high-level architecture.

III. MixLock

A. Principle of operation

MixLock constructs a locked M-S IC by locking its digital section, as shown in Fig. 2. A M-S IC is composed of several analog, mixed analog-digital, and purely digital cores that are interconnected in a serial signal-processing chain and/or in feedback loops. By locking one or more digital cores, we gain control over the signal-processing flow. According to the M-S IC type, the digital cores can perform different whole functions or subfunctions. Using an incorrect key, the output of the digital core(s) will be corrupted, which will in turn affect the signal-processing flow and corrupt the M-S IC performance trade-off in a complex and unpredictable way; i.e., one or more performances will lie outside their acceptable specification range. In short, logic locking of the digital section ensures performance degradation of the M-S IC on application of an incorrect key.

B. Attributes

1) Wide applicability: The *MixLock* principle applies at system-level for large M-S ICs that comprise several blocks, some of which are mixed analog-digital or purely digital. This includes a wide range of M-S ICs, including data converters, Phase-Locked Loops (PLLs), and RF transceiver architectures. It also fits well the general trend towards digitally-assisted analog designs and digital centric M-S architectures, where the goal is to make a thoughtful shift of functionality from the analog into the digital domain, in order to alleviate analog design complexity and enable post-manufacturing tuning, self-calibration, and re-configurability.

More specifically, all ADCs comprise digital blocks that can be targeted for locking. $\Sigma\Delta$ ADCs, which will serve as our case study, comprise a $\Sigma\Delta$ modulator and a digital decimation filter used for removing the out-of-band noise from the $\Sigma\Delta$ modulator output and down-sampling it to convert it to a high-resolution digital signal, sampled at the Nyquist rate. Successive Approximation Register (SAR) ADCs comprise several mixed analog-digital and purely digital blocks, including Digital-to-Analog Converters (DACs) that create the comparison levels and the SAR logic block that controls the conversion process. Pipeline ADCs comprise digital logic that assembles the digital codes of the cascaded stages and performs the digital correction to reduce the accuracy requirement of the flash sub-ADCs used inside each stage. Regarding PLLs, one can target locking the Voltage Controlled Oscillator (VCO) or the frequency divider. Regarding RF transceivers, the most common architectures, i.e., Zero Intermediate Frequency (Zero-IF), Low Intermediate Frequency (Low-IF), and super-heterodyne, are based on I/Q branches and both the receiver and transmitter require a digital block to correct for DC offset and/or I/Q imbalance. In this case, locking can be performed on this block to create artificial I/Q imbalance and corrupt the Bit Error Rate (BER). Regarding highly-digitized RF transceiver architectures, the receiver uses an RF ADC to directly convert the RF signal to the digital domain, which is then down-converted to DC by a digital mixer and filtered using a digital decimation filter. In the transmitter, the baseband signal passes through a digital interpolation filter before the digital up-conversion. Locking here can target the digital decimation and interpolation filters.

2) Non-intrusiveness: In fact, the main advantage of *MixLock* is that it achieves locking of the M-S IC while deliberately retaining intact the analog cores without needing to re-design them. Adding any lock mechanism inside the analog cores is likely to find analog designers reluctant since it inevitably degrades performance and consequently requires re-designing the analog cores together with the lock mechanism so as to meet the intent specifications, if this is possible at all. The fact that *MixLock* is transparent and non-intrusive to the analog cores makes it very attractive for analog designers and is vital for its wide adoption in the scope of a M-S design flow.

3) Large key size: The fact that *MixLock* inserts the lock mechanism inside the digital sections allows using a large key size which is a prerequisite for thwarting counter-attacks on locking that search for the secret key. Such a large key size is arguably difficult to be introduced into the analog cores without significant overheads and re-design effort.

4) Unaltered analog design flow and full automation: Another important advantage of *MixLock* is that it leaves intact the analog design flow, which is not automated and involves manual schematic and layout design, as well as manual layout floor-planning and routing. The M-S IC design flow with *MixLock* alters only the digital design flow and is fully automated, since logic locking is fully automated, involving a few additional steps after logic synthesis. The locking step can be seamlessly integrated into the digital design flow.

5) Low-overhead: The area and power overheads of *MixLock* are those of the locked digital section projected to the complete M-S IC. Typically, die area and power consumption in a M-S IC is largely dominated by the analog section. The area and power overhead in the digital section introduced by logic locking is already affordable considering the digital section alone; this overhead, when projected to the entire M-S IC, will be even easier to justify.

Furthermore, advanced logic locking algorithms and TRLL in particular [5] allow the designer to designate critical path locations in the digital design where no logic modifications are to be carried out. Ideally, the lock insertion is to be performed in non timing-critical paths with sufficient timing slack. Thereby, timing violations can be avoided and, in return, any small

delay penalty in the digital section due to the insertion of the lock mechanism will be non-critical for meeting the M-S IC performance trade-off. For our case study circuit in Section IV-A, *MixLock* did not induce any performance degradation in the M-S domain. We expect that *MixLock* will generally only induce minor performance penalties.

6) Independence from logic locking: *MixLock* is independent of the underlying logic locking mechanism and can make use of the state-of-the-art logic locking technique at any given point in time. This is important to clarify since, as we will discuss in Section III-D, there have been a plethora of logic locking techniques which shortly after their appearance were broken by a counter-attack. Our current *MixLock* 2.0 implementation makes use of the state-of-the-art logic locking technique, namely DisORC in combination with TRLL [5], which is resilient to all known attacks on logic locking.

It should be mentioned, however, that in general logic locking techniques cannot be blindly applied in this context and they need to be fine-tuned so as to effectively break the performance of the M-S IC when incorrect keys are applied. This point will be discussed in more detail in Section III-E.

7) Resilience to attacks: As we will discuss in detail in Section III-G, the *MixLock* 2.0 implementation is capable of co-optimizing security in the analog and digital domains.

C. Security metrics

The security level offered by *MixLock* is defined in terms of security level of the underlying logic locking, called *digital security level*, and the security level of M-S performance trade-off locking, called *analog security level*.

The digital security level is measured by the effort that the designer must spend for identifying the secret key. It is dictated by the resilience against known counter-attacks, as will be explained in more detail in Section III-D.

The analog security level is measured using three different metrics in the analog domain:

1) *Percentage of failing keys*. The percentage of incorrect keys resulting in violation of one or more performance specifications. This metric may be misleading if incorrect keys only slightly push performances outside their specifications. To account for this scenario, we also use the following two metrics.
2) *Mean error*. The average performance difference between the unlocked M-S IC and the locked versions with incorrect keys.
3) *Minimum error*. The minimum observed performance difference between the unlocked M-S IC and the locked versions with incorrect keys, indicating the *worst-case* locking scenario.

These metrics can be quantified by putting to test a large set of random incorrect keys.

D. Short history on logic locking and counter-attacks

Initial logic locking solutions, such as Random Logic Locking (RLL) [24], Fault analysis-based Logic Locking (FLL) [25], and Strong Logic Locking (SLL) [26], were based on inserting key-controlled logic gates within the circuit and aimed at achieving high output corruption, i.e., a heavily non-functional circuit, on application of an incorrect key. However, such a strategy was outlived by the miter-based SAT attack [27]. It recovered the secret key by pruning out multiple key-values per iteration. In the process, the attack queries a working chip (also referred to as an *oracle*) with carefully crafted input patterns, and also makes use of the existing scan infrastructure on the chip.

To overcome this attack strategy, researchers improved SAT resilience of logic locking techniques based on key-gate insertion [28] and also proposed point-function based locking techniques which pushed the limits of the oracle-based SAT attack to brute-force by eliminating exactly one key-value per iteration [7]. The latter techniques faced the drawback of low output corruption on incorrect key assignment. Following this observation, a merger between pre-SAT high output corruption locking schemes and post-SAT low output corruption locking schemes was proposed [29]. But the limitations of either of the schemes paved way for advanced and tailored oracle-based attacks [30].

The reason why oracle-based attacks are successful even on large sequential digital designs is because of the presence of scan-chains. Scan chains are necessary for thorough testing of the chips; they allow direct access to the internal nodes in the design without relying on multiple capture cycles. This construction within the design is utilized by SAT-based attacks to successfully and efficiently recover the secret key.

Still, there are oracle-based attacks that do not rely on scan chains and aim at recovering the secret key only by granting primary I/O access, such as the KC2 attack [31]. The KC2 attack unrolls the combinational part of the sequential circuit by creating multiple copies of it. The number of copies that the attack needs to create depends on the sequential depth of the design.

Another category of attacks is based on netlist or structural analysis. They inspect the locked netlist with the aim of deciphering the key-values from the structure and type of embedded logic gates. These attacks are referred to as oracle-less attacks as they do not rely on a working chip. Structural analysis-based attacks using machine learning [32], [33] undo the obfuscation realised by synthesis tools and map the logic gate type directly to its key-value. Other oracle-less, netlist-only attacks include the redundancy attack [34] which inspects the netlist to identify redundant nodes on application of random key-values. If any key-value on the key-gate introduces redundancy in the netlist, then that key-value is discarded.

E. Choice of logic locking in the context of MixLock

In the context of *MixLock*, the chosen underlying logic locking technique should thwart all existing oracle-based and oracle-less attacks (digital security) all-the-while achieving high output corruption for incorrect keys (analog security).

For various reasons, M-S ICs do not always include scan chains into their digital section, even when the amount of logic is large. For example, the M-S IC will be, after all, tested as a whole; inserting scan chains affects speed; and many analog

designers are not familiar with scan chains. If scan chains are absent, oracle-based attacks requiring scan access, such as the original SAT attack [27], are disabled. Then, to attain digital security, a designer should be mindful of oracle-less attacks that analyze the reverse-engineered netlist for clues about the secret key [32], [33] and of oracle-based attacks using I/O data and no scan access, such as the KC2 attack [31]. The objective is to select the smallest-overhead logic locking technique that achieves strong analog security while digital security is also ensured. If scan chains are present, then achieving digital security requires more effort to thwart both oracle-based and oracle-less attacks, and thus, a more sophisticated logic locking technique should be used.

The previous *MixLock 1.0* implementation [6] utilized the SFLL technique [7] which thwarts only oracle-based attacks. In brief, SFLL hard-codes a comparison $\text{HD}(IN, \text{key}) = h$, where $\text{HD}(x, y)$ is the HD between x and y and IN is the digital input pattern to the locked block. If the comparison does hold true, then a victim bit is flipped. A correction unit is used to implement the same comparison, but this time the key is sourced from the TPM. When a bit-flip happens, the correction unit will immediately flip again the bit upon application of the correct key, thus the error will be suppressed. SFLL produces incorrect functionality only for a small set of PIPs that have a HD=h from the key, when an incorrect key is applied. By increasing h we can increase functionality corruption, but this is at the expense of lower security levels against the SAT attack [7]. New research shows that this trade-off doesn't necessarily hold true and higher corruption rates can be achieved without sacrificing security [35], [36]. However, the area overhead for a high-security/high corruption locking may be large.

Still, low corruption rates are oftentimes sufficient for digital ICs. For example, to lock a microcontroller it suffices to lock one bit for one input in the program counter to safeguard against unauthorized execution [7]. However, in the context of M-S IC locking, flipping a bit inside a digital block for a small set of its inputs may not result in any appreciable performance degradation. For example, in [6], SFLL was used to lock a $\Sigma\Delta$ ADC via locking the most-significant bit (MSB) output of the first comb filter within the decimation filter. The analog input propagated to the input of the locked decimation filter rarely "hits" one of its PIPs. This essentially means that the performance will be corrupted under specific input conditions which are not necessarily met frequently. A suitable secret key was crafted so as to achieve the maximum possible functionality corruption while still achieving at least $64\,\text{bit}$ resilience against the SAT attack. To this end, to further increase the functionality corruption, an additional second instantiation of SFLL, locking the MSB-1 of the comb filter, was used. This second lock mechanism increases functionality corruption but has a low resiliency against SAT attack and, thereby, can be broken, leaving the device protected with only the first lock mechanism. This variant of SFLL with two lock mechanisms is referred to as 1.5xSFLL in the *MixLock 1.0* implementation.

In our new *MixLock 2.0* implementation presented in this paper, we make use of the recently proposed state-of-the-art DisORC technique in combination with TRLL [5]. This technique heavily benefits *MixLock* as it leads to very high functionality corruption and, accordingly, a very strong analog security level, while achieving low area overhead and protecting against all known logic locking counter-attacks, thus delivering digital security as well.

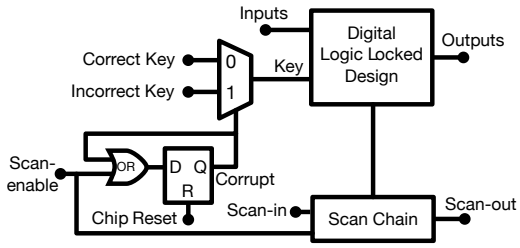


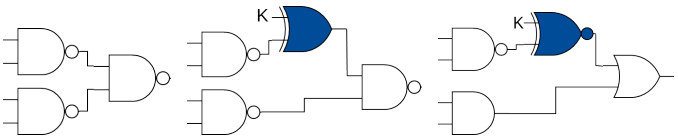
Fig. 3: DisORC defense architecture.

F. DisORC with TRLL

DisORC and TRLL [5] are two orthogonal solutions, meaning that DisORC provides total resilience against oracle-based attacks that make use of the scan chain [27] while TRLL provides maximum functionality corruption and total resilience against oracle-less attacks based on structural analysis using machine learning [32], [33].

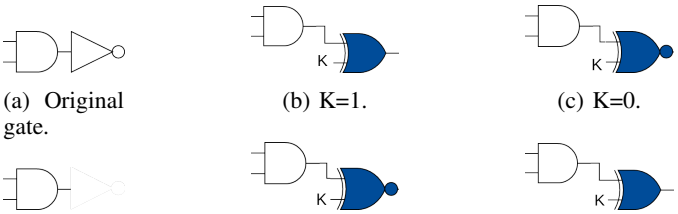
Fig. 3 presents a high-level description of the DisORC defense architecture. The DisORC principle is to withdraw the secret logic locking key upon detection of an access to the scan chains, while maintaining full testability of the circuit. More specifically, the key feeding the key-gates is controlled by a new signal called *corrupt* that is generated out of the scan-enable signal, which denotes scan access. In functional mode, scan-enable is always set to 0, thereby allowing the correct key to be fed to the key-gates. In test mode, where correct functionality of the circuit is not required to attain structural test coverage, scan-enable is possibly exercised by the chip user to load and unload scan chains. This sets the corrupt signal to a permanent 1 until a chip power-reset, disconnecting the secret key from the design and instead allowing the chip user to choose any dummy incorrect key to drive the key-gates. A high fault coverage can still be attained as the keys act as test points. Authentic users can also load the correct secret key, but through the test interface instead, to perform scan-assisted functional debug. Unauthentic users cannot load/unload the scan chains while the secret key drives the design, thus scan access is not helpful to them in retrieving the secret key.

Further, the DisORC defense needs to be incorporated with another layer of logic locking to ensure good output corruption, which is of utmost importance in the context of *MixLock*. Traditional logic locking defenses, such as RLL [24], FLL [25], and SLL [26], deliver such corruption, but recently, machine learning-based oracle-less attacks were proposed on these schemes, recovering the key-value from the netlist without the need of any oracle [32], [33]. The underlying reason is that the key-gate type (XOR or XNOR) implies the key-value (0 or 1, respectively) even though inverter/bubble pushing transformations are performed by the synthesis tool on the netlist after inserting the key-gates. Fig. 4 illustrates



(a) Original netlist. (b) Naive RLL; K=0. (c) RLL after bubble pushing; K=0.

Fig. 4: RLL using XOR key-gate.



(a) Original gate. (b) K=1. (c) K=0.

(d) Original gate. (e) K=1. (f) K=0.

Fig. 5: TRLL application on logic gates.

these transformation operations for an XOR key-gate. A naive integration of these locking schemes with DisORC makes them susceptible to such oracle-less attacks.

DisORC is instead integrated with the TRLL defense, which thwarts oracle-less attacks. TRLL inserts key-gates or replaces existing gates (inverters) with key-gates randomly in the netlist. As shown in Fig. 5, an existing inverter in the design may be replaced by a key-gate (Figs. 5(a)-(c)) or a key-gate may be inserted to a design (Figs. 5(d)-(f)), negating the key-value polarity. Without knowing the original gate structure, the attacker cannot differentiate the designs in Figs. 5(b) and 5(f) and the designs in Figs. 5(c) and 5(e), thereby failing to identify the correct key-value. As these decisions are made randomly by TRLL, the attacker has no way of knowing or learning the key-value from the key-gate type or gate structures surrounding the key-gates.

G. Attack resiliency in MixLock 2.0

We assume the most demanding threat model where the attacker has access to an oracle with the correct key stored in the TPM and also possesses the netlist of the locked design.

1) Brute-force and optimization attacks: Brute-force is the most straightforward attack involving trials of random keys by simulating the design at transistor-level or layout-level, hoping to identify a key that establishes correct functionality, that is, a desired performance trade-off within the specification range. For a key with k key-bits, the search space is 2^k. Alternatively, to guide the search the attacker can use optimization algorithms, such as gradient descent, simulated annealing, and GAs, hoping to converge fast to the key.

Defense. *MixLock 2.0* is resilient to such attacks since: (a) it allows introducing a large key size k (e.g., $k = 128\,\text{bit}$ in our case study), resulting in a very large key space; (b) simulating all the test benches of a M-S IC at transistor-level to measure the performances can be extremely time-consuming (e.g., in the order of hours for the $\Sigma\Delta$ ADC case study), thus the attacker in practice can afford carrying out just a few trials; and (c) using DisORC with TRLL we achieve high functionality corruption, thus there are no, or only a very limited number, of incorrect keys that could establish approximate functionality. Essentially, the desired performance trade-off behaves as a delta function on the correct key, thus the optimization is likely to "zigzag" endlessly.

2) Attacks on logic locking: Main attacks include oracle-based attacks utilizing SAT and scan access [27], oracle-less attacks based on netlist analysis using machine learning [32], [33], and the oracle-based KC2 attack [31].

Defense. As discussed in Section III-F, by construction DisORC thwarts oracle-based attacks utilizing SAT and scan access and TRLL thwarts oracle-less attacks based on netlist analysis using machine learning. The KC2 attack is the only oracle-based attack that can be applied. However, it is not scalable for large circuits and, in addition, it assumes I/O access, which may not be granted in a M-S IC. The reason is that I/O nodes of the target digital sub-block for locking may be highly sensitive, e.g., high frequency nodes shared with I/O of analog sub-blocks. Adding parasitics or wire load to a high frequency node, for instance by routing it to a pad, would greatly degrade the M-S IC performance and is therefore prohibitive. We launched the KC2 attack on our case study and it did not succeed as we demonstrate in Section IV-D.

3) Removal attack: It traces the key-bits structurally and aims at removing the lock.

Defense. Removing only the DisORC component, which can be easily identified, leaves the TRLL key-gates intact in the design, thus the removal attempt is incomplete. TRLL prevents identifying the functionality of the locked digital section, thus the removal attack cannot be completed. Even if the functionality is known, removing the digital section entirely and replacing it with a re-design is not straightforward. The reason is that digital cores are closely intertwined with the analog cores and redesigning them requires knowledge about the interfaces between A/M-S and digital cores, e.g., voltage levels, data synchronization procedures, or sampling frequency ratios, that go way beyond the information published in datasheets. This level of knowledge and the fact that digital section development may take several months of work even for expert designers render a re-design attack prohibitively expensive. Therefore, an attacker is left only with the analog cores which serve for nothing without their digital counterparts.

4) Attacks proposed for biasing locking: SMT-based and GA-based attacks targeting biasing locking were discussed in detail in Section II-B.

Defense. SMT-based and GA-based attacks assume the existence of an analog obfuscated component, e.g., the mirror transistor in a current mirror. Therefore, these attacks are non-applicable to *MixLock*.

5) Attack on mixed-signal locking: The attack on mixed-signal locking proposed in [17] was described in Section II-B.

Defense. This attack concerns only SFLL that corrupts the output for a handful of PIPs. The *MixLock 2.0* implementation employs TRLL, for which all input patterns to the locked

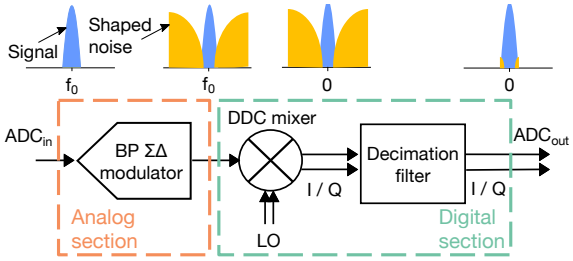


Fig. 6: Architecture of the RF ΣΔ ADC.

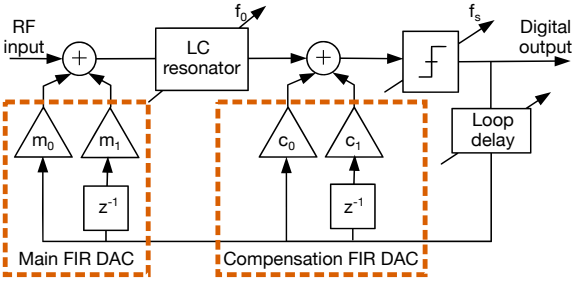


Fig. 7: Architecture of the ΣΔ modulator.

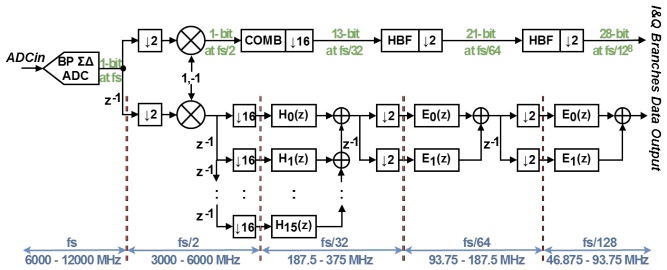


Fig. 8: Architecture of the decimation filter of the ΣΔ ADC.

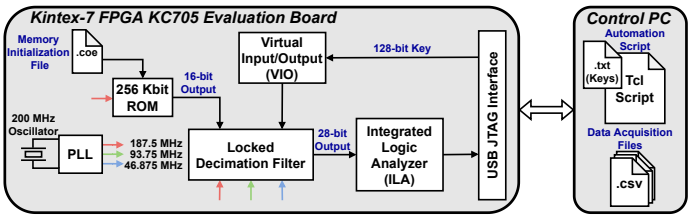


Fig. 9: Implementation of the digital section of the ΣΔ ADC on a Xilinx Kintex-7 FPGA.

digital section are PIPs. Thus, finding the PIPs, which is one of the steps in the attack in [17], is not possible.

IV. Hardware Experiment

A. Case study

We use as case study a state-of-the-art bandpass RF ΣΔ ADC implemented in a 65 nm CMOS process [37]. The high-level architecture is illustrated in Fig. 6. The ADC has a tunable center frequency f_0 from 1.5 GHz to 3 GHz and a corresponding sampling frequency f_s from 6 GHz to 12 GHz, converting a band of 47 MHz and 93 MHz centered around f_0, respectively.

The analog section of the ΣΔ ADC is a 2nd order LC bandpass ΣΔ modulator, illustrated in Fig. 7. The modulator oversamples the analog input signal and generates a high-frequency, low-resolution 1 bit digital signal at its output.

The 1 bit digital output is interfaced with the ΣΔ ADC's digital section, which consists of a digital down-conversion (DDC) mixer and a multi-stage multi-rate decimation filter, illustrated in Fig. 8. The decimation filter removes the out-of-band noise from the ΣΔ modulator output and down-samples it by a factor of 64 to the baseband. Thereby, it provides a high resolution 28 bit Nyquist rate sampled digital signal at its output. Structurally the decimation filter is composed of three subfilters. The first being a comb filter (COMB) followed by two half-band filters (HBF1 and HBF2) with down-conversion ratios of 16, 2, and 2, respectively. The DDC mixer adds a supplementary down-conversion by a factor of 2, so the total down-sampling factor is $64 \cdot 2 = 128$.

To perform the experiment we use the ΣΔ modulator of the actual chip [37], while the digital section is implemented on a Xilinx Kintex-7 FPGA operating at a frequency of 187.5 MHz in order to facilitate the experiment, i.e., try out different locked versions and keys. The register-transfer level (RTL) design is first synthesized using the Nangate 45 nm open source cell library so as to insert the lock at gate-level. The locked gate-level design is then implemented in the Xilinx Kintex-7 FPGA using Xilinx's Vivado Design Suite. Fig. 9 shows the FPGA implementation's architecture. The ΣΔ modulator output bit-stream is saved into the Read-Only Memory (ROM) of the FPGA. Fetching the key, as well as the acquisition of the 28 bit decimation filter output, is achieved via a JTAG-USB interface with the control PC. A PLL generates the three different clock frequencies required for the decimation filter's sub-stages.

The goal of the case study is to lock the main performance of the ΣΔ ADC, namely its Signal-to-Noise Ratio (SNR). This is achieved via locking of the decimation filter by leveraging DisORC with TRLL. In particular, TRLL is applied to lock the first sub-filter stage of the decimation filter, i.e., the comb filter, so as to make errors spread out down in the signal processing chain. We implement a 128 bit secret key which, in contrast to the work in [6], is not crafted to present a sufficient amount of errors but instead is chosen at random. The ΣΔ ADC has a nominal SNR of 34.8 dB with a specification set at 30 dB. Any incorrect key should break the SNR performance, rendering the ΣΔ ADC unusable.

B. Experimental setup

To allow for the generation of comparable results from hundreds of repetitions of the same experiment in which different keys are applied, we record the ΣΔ modulator chip's high-frequency output bit-stream for a defined input waveform. In particular, we use a sinusoidal input with frequency $f_{\text{in}} = f_0 + \Delta f = 1.5\,\text{GHz} + 300\,\text{kHz}$, corresponding to a sampling frequency f_s of 6 GHz. We then record 524 288 samples of the output bit-stream, corresponding to 12 full input signal periods. Subsequently, to interface the ΣΔ modulator chip's output to the FPGA, we down-convert it to the baseband with the help of a DDC mixer implemented in Simulink. This down-converted bit-stream is transformed to a memory initialization

file and saved in the ROM of the FPGA. In each repetition of the experiment, the ROM provides this same bit-stream to the decimation filter with embedded locking.

A single experiment to try one key takes approximately 1.2 seconds and consists of the following steps:

1) The FPGA loads a key via the USB interface and applies it to the locked decimation filter.
2) The decimation filter processes the pre-recorded output bit-stream of the ΣΔ modulator which is retrieved from the ROM memory.
3) The integrated logic analyzer acquires 4096 samples from the decimation filter's output to export them as a comma-separated values (CSV) file to the control PC via the USB connection.
4) On the control PC a Fast Fourier Transform (FFT) is performed on the captured data and the SNR is derived.

C. Results and security analysis

The security metrics proposed in Section III-C are evaluated first for 10^3 randomly generated incorrect keys. We achieve the strong analog security metrics shown in the first row of Table I. The percentage of failing keys is $100\,\%$, that is, any random key breaks the circuit's functionality. The mean SNR of failing keys is $-15.4\,\mathrm{dB}$, the mean error is $45.4\,\mathrm{dB}$, and the minimum error is $35.1\,\mathrm{dB}$, computed as a distance from the specification of $30\,\mathrm{dB}$. Fig. 10 shows the SNR performance of the ΣΔ ADC for the 10^3 incorrect keys and the correct key. The unlocked ΣΔ ADC stands out with a correct SNR of $34.8\,\mathrm{dB}$. Locked versions have an SNR well below the specification of $30\,\mathrm{dB}$, actually below the noise level. This shows that unless the correct key is provided, the input signal gets completely buried under the noise floor.

Given that M-S ICs allow a large margin for their performances, it is likely that there exist incorrect keys that establish correct functionality. In the *MixLock 2.0* implementation, such keys will have a small HD from the actual secret key. To approximate the space of such keys, we evaluate incorrect keys starting with HD=1 and increasing the HD until a value for which all keys fail. For HD=1, we evaluated all 128 keys, whereas for HD>1, we evaluated 10^3 randomly generated keys. As shown in Table I, for HD=1 74.42% of the keys fail. The percentage of failing keys increases with HD and beyond HD=7 all keys fail. We can estimate the number of passing incorrect keys as $\sum_{i=1}^{n=6}\left(1-\frac{w_i}{100}\right)\cdot\binom{128}{i}$, where w_i is the percentage of failing keys for HD$=i$ and n is the highest HD showing passing keys. This number equals approximately 10^{20}, thus the nominal key width of $128\,\mathrm{bit}$ is reduced to 108 effective bits, still a very strong analog security level.

Table I also provides the security metrics for the *MixLock 1.0* implementation that employs SFLL. By construction of the SFLL lock mechanism, for any incorrect key having HD > 1 from the correct key, every query of the circuit with the same input waveform leads to the same errors and, thereby, the SNR performance degrades by the same amount. As it can be seen from Table I, the *MixLock 2.0* implementation results in higher functionality corruption for random incorrect keys, as shown by the larger mean and minimum SNR error.

TABLE I: Analog security metrics for invalid keys with different HD from the correct key computed with respect to the SNR specification.

Hamming Distance	Percentage of failing keys	Mean error	Minimum error
MixLock 2.0			
random	100%	45.4 dB	32.4 dB
1	74.42%	14.6 dB	0.2 dB
2	93.92%	18.6 dB	0.2 dB
3	98.30%	21.8 dB	0.4 dB
4	99.60%	24.2 dB	0.4 dB
5	99.90%	25.8 dB	1.0 dB
6	99.98%	28.0 dB	2.0 dB
7	100%	29.5 dB	2.3 dB
MixLock 1.0			
any ≥ 1	100%	22.3 dB	22.3 dB

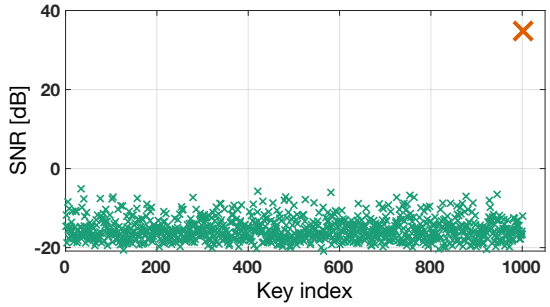


Fig. 10: SNR of the ΣΔ ADC for 10^3 random keys. The cross corresponds to the SNR of the unlocked ΣΔ ADC.

Finally, Fig. 11 illustrates the transient and frequency responses recorded at the ΣΔ ADC's output, where a random incorrect key leads to important levels of distortion in the time domain, while in the frequency domain this distortion is visible in the form of an important increase in the noise level.

D. Attacks on logic locking

The decimation filter is a sequential design with 30K gates and scan testing is a highly desirable feature. As explained in Section III-G, by construction DisORC with TRLL thwarts all attacks except the KC2 attack. The KC2 attack requires access to the I/O of the digital section of the ΣΔ ADC which, however, is not provided in a full ASIC implementation. This is because the input of the digital section is also the high-frequency ΣΔ modulator's output. Routing this node to a pad would greatly degrade the ΣΔ ADC's performance and is therefore prohibitive. Nevertheless, we set up the KC2 attack to evaluate its capability. Even when granting KC2 50 GB of memory and 48 h of runtime it fails to recover a functional key for the locked ΣΔ ADC.

E. Implementation costs

Table II presents the overheads that *MixLock 2.0* induces with regard to the nominal ΣΔ ADC design. Overheads are reported for an ASIC implementation of the ΣΔ ADC and thus also an ASIC implementation of the decimation filter,

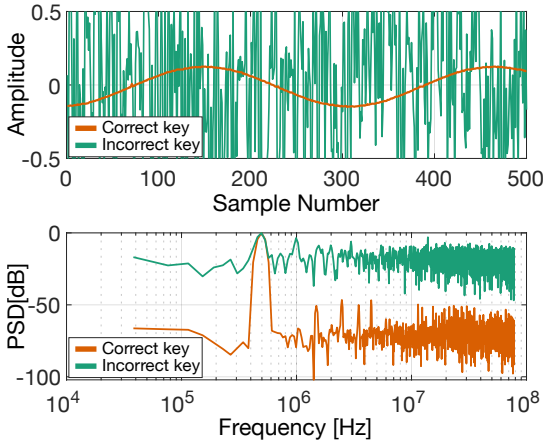


Fig. 11: Measured output of the locked ΣΔ ADC with the top plot showing a zoom of the output in the time domain and the bottom plot showing the Power Spectral Density (PSD).

TABLE II: *MixLock 2.0* overheads.

Decimation filter	Overhead	ΣΔ ADC	Overhead
Area [%]	10.7	Area [%]	3.6
Power [%]	21.3	Power [%]	7.1
Delay [%]	13.4	Performance [%]	0.0

synthesized using the Nangate 45 nm library. On the left hand side the overheads with respect to only the ΣΔ ADC's digital section are given, on the right hand side the overheads are projected to the entire ΣΔ ADC considering that the digital section occupies about 30% of the die area and is responsible for about 30% of the total power consumption. The slack reserve in the critical path of the digital section is large enough to accommodate the additional key-gates; thus, the added delay gets easily absorbed and does not translate to an SNR performance penalty. The unlocked ΣΔ ADC has an SNR of 34.8 dB, that is, there is no performance degradation due to locking. The area and power overheads of 10.7% and 21.3% for the digital section translate to area and power overheads of 3.6% and 7.1% for the complete ΣΔ ADC, which are very reasonable.

V. Audio Demonstrator

A. Demonstrator configuration and setup

To understand the impact of locking on real-world analog signals, we use an audio demonstrator originally presented in [38]. Herein, we show results obtained using the *MixLock 2.0* implementation, while results obtained using the *MixLock 1.0* implementation are provided in [38]. A high-level description of the audio demonstrator is illustrated in Fig. 12. In a first step we use Matlab to read in an audio waveform, either from an existing audio file or by recording it with a microphone via the function `audioread`. The audio is upsampled using linear interpolation between its data-points so as to provide enough data samples for the subsequent second digitization by the oversampling ΣΔ ADC. The ΣΔ ADC's output is written to the hard-disk as an audio file using the `audiowrite` function to then be played back through the PC's speakers. The

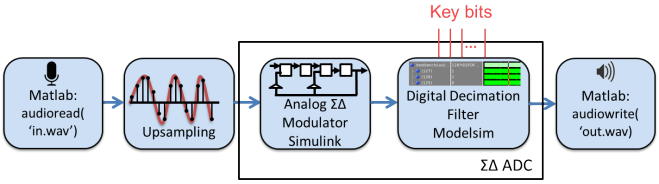


Fig. 12: Block diagram of the audio demonstrator for *MixLock*.

ΣΔ modulator in this demonstrator is a system-level model of a second-order low-pass continuous-time ΣΔ modulator. The modulator is modeled and simulated in Simulink, while the decimation filter is modeled in RTL and is simulated in Modelsim.

Our experiment involves processing the audio samples listed in the first column of Table III through the system in Fig. 12 and examining the effect of locking on the audio quality. Audio samples include a speech recording in English and professional music recordings of various genres. Similar to Section IV, TRLL is applied to the comb filter of the decimation filter with a 128 bit key. All audio samples are evaluated using the correct key and four incorrect keys with HDs of 4, 8, 12 and 63 with respect to the correct key. While the first three incorrect keys are favorably chosen for the attacker, the latter key with a HD of 63 corresponds to a random key.

Herein, the SNR metric cannot be used for quantifying analog security as it was used in Section IV. The reason is that SNR requires a sinusoidal input, while audio signals are time-varying in nature; their spectral contents vary with time and they are rich in frequencies. For this purpose, we use Root-Mean-Square Error (RMSE) calculated over the complete duration of the audio file as metric to evaluate the error between two waveforms processed by an unlocked and a locked ΣΔ ADC for the same input.

While RMSE allows to quantify the error in the processed audio samples, it does not allow us to understand in what kind of way an audio sample is impacted, i.e., two RMSE values with similar magnitude may be derived from erroneous audio samples that *sound* differently. Even so, unless the secret key is applied to the ΣΔ ADC, locking introduces errors that get translated into audible glitches or noise that deteriorate the output signal, which can be measured via the RMSE metric.

B. Demonstrator results

The reader can download and listen to the output audio samples via this link: https://nuage.lip6.fr/s/j6F54tWqHgsdBGL. The downloadable archive includes the output audio samples listed in Table III for an unlocked design when the valid key is applied, as well as for a locked design using TRLL when incorrect keys with HDs 4, 8, 12 and 63 from the correct key are applied. The archive also includes the output audio samples using the methods SFLL and 1.5xSFLL [6]. The fourth to last columns of Table III show the RMSE of the audio samples for the locked designs using TRLL, whereas for the correct key the RMSE metric is constantly zero.

TRLL leads to a corruption of the audio output in a way that is comparable to white noise. The higher the provided

TABLE III: Audio samples and impact of locking with *MixLock 2.0* on audio quality measured in RMSE between the nominal waveform and the respective waveform of the locked ΣΔ ADC.

Audio sample	Duration	Sample Rate	RMSE using DisORC with TRLL			
	[s]	[Hz]	HD=4	HD=8	HD=12	HD=63
English voice recording	4	8192	0.184	0.530	0.341	0.379
Bob Marley - No Woman No Cry	15	16384	0.035	0.058	0.184	0.354
Benny Goodman - Bugle Call Rag	15	16384	0.024	0.041	0.156	0.333
Kenny Ball - I Wanna Be Like You	15	16384	0.025	0.043	0.167	0.335
John Coltrane - Nature Boy	15	16384	0.026	0.044	0.161	0.349
Beethoven - Symphony No. 9	15	16384	0.024	0.041	0.177	0.298

key's HD from the correct key is, the greater the impact of the corruption is. 1.5xSFLL also corrupts the audio quality dramatically distorting the recording with very frequent glitches. SFLL on the other hand results in a number of glitches for the music recordings that can be heard as single, loud "cracks". However, for the self-made voice recording, no glitches are noticed.

Furthermore, given that the locations where TRLL modifies the comb filter's gate-level netlist are chosen randomly, the errors that are induced due to an incorrect key are not restricted to the MSB or MSB-1 such as was the case for both SFLL variants. Errors in fact appear anywhere in the comb filter circuit, thereby inciting faults of reduced scale, albeit in very elevated numbers. While both SFLL and 1.5xSFLL are independent of the user provided key, for TRLL we observe a dependence of the RMSE metric on the key. The closer a key is to being correct, the fewer errors are induced, as shown in Table III.

Fig. 13 visualizes the effect of locking using TRLL in a transient waveform on the example of a short sequence of Beethoven's Symphony No. 9. While still roughly following the correct waveform, the locked waveform is erroneous in practically every single sample point. The listener perceives these errors as dominant and disturbing white noise overlaying the distantly recognizable remains of the original waveform.

At this point it is important to recall the objective 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 random noise. The aim is to render the hardware low-quality and unusable unless the valid secret key is known; e.g., glitches occurring at regular and frequent intervals are sufficient.

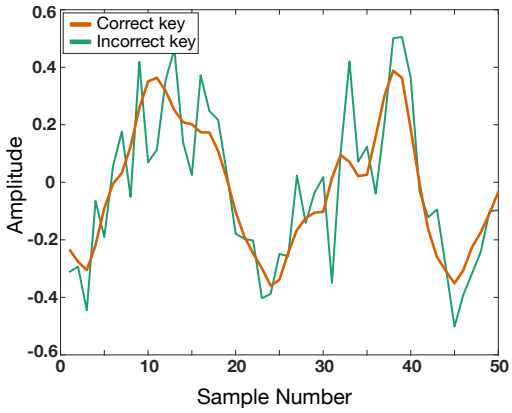


Fig. 13: A short sequence of Beethoven's Symphony No. 9 shown as transient waveform processed by an unlocked and a locked ΣΔ ADC.

VI. *MixLock* COMPARISON TO OTHER A/M-S IC LOCKING TECHNIQUES

Table IV compares the existing A/M-S IC locking techniques based on different criteria, namely applicability, overhead (i.e., performance, area, and power penalties), design complexity, attack resilience, and protection against cloning. All techniques are effective against other types of counterfeiting, thus for brevity these threats are not included in Table IV.

Biasing locking is the most general technique that applies also to purely analog blocks. Calibration locking requires the existence of a calibration mechanism that acts on tuning knobs judiciously inserted into the design. The technique in [13] is the most general since most on-chip calibration schemes comprise a digital processor. The technique in [14] is the most powerful but assumes high programmability and the off-chip calibration algorithm must be complex enough such that it cannot be devised by the attacker. AFGTs are not often used for calibration, but the technique in [15] opens a new application domain. *MixLock* is generally applicable to any M-S IC, including data converters, PLLs, and RF transceivers, as discussed in Section III-A.

In terms of area overhead, biasing locking techniques based on biasing circuit expansion [8]–[10] feature low area overhead, while memristor-based [11] and neural network-based [12] biasing require large on-chip resources. In [8], a 6% area overhead is reported when applying biasing locking to a PLL based on transistor width obfuscation. M-S locking techniques that leverage logic locking, i.e., [13] and *MixLock*, also result in justifiable area overhead. *MixLock* applied to a ΣΔ ADC results in 6.7% and 8.1% area overhead respectively when using as the underlying logic locking technique SFLL and 1.5xSFLL [6], while when using DisORC with TRLL the area overhead is 3.6% as shown in Table II. In [13], the reported area overheads by locking the digital optimizer in the calibration loop for different case studies including a band-pass filter, low noise amplifier, and low-dropout regulator, are in the range of 2.6% to 8.8%. The extra on-chip resources

TABLE IV: Comparison of different A/M-S IC locking techniques.

	Applicability	Overhead	Design complexity	Attack resilience	Anti-Cloning
Biasing locking					
Transistor width obfuscation [8]	general	low	low	X	✓
Current mirror locking [9]	general	low	low	X	✓
Mesh transistor obfuscation [10]	general	low	low	X	✓
Memristor crossbar [11]	memristor technology	high	high	X	✓
On-chip neural network [12]	general	high	high	X	✓
Calibration locking					
Calibration loop logic locking [13]	tuning knobs on-chip digital calibration loop	low	low	✓	✓
Highly-digitized A/M-S ICs [14]	tuning knobs complex off-chip calibration algorithm	no	no	✓	✓
Limiting tuning range of AFGTs [15]	AFGT-based programming	low	low	X	X
***MixLock* (this work)**	mixed-signal	low	low	✓	✓
Compound locking [16]	mixed-signal	medium	low	✓	✓

for locking the tuning range of AFGTs should also present a small area overhead, although it is not reported in [15]. Using the actual programmability fabric to enable locking has zero overhead [14].

In terms of power overhead, biasing locking based on biasing circuit expansion [8]–[10] seems to provide the lowest power overhead. In [8], a 1% power overhead is reported when applying biasing locking to a PLL based on transistor width obfuscation. For the calibration locking techniques in [13], [15] that use an on-chip calibration scheme, power overhead does not apply as the calibration scheme is only active at startup. For the calibration locking technique in [14], there is no lock, thus there is no extra power consumption. *MixLock* offers justifiable power overhead. When applied to a $\Sigma\Delta$ ADC, *MixLock* results in 9.8% and 11.8% power overhead respectively when using SFLL and 1.5xSFLL, while when using DisORC with TRLL the power overhead is 7.1% as shown in Table II.

In terms of performance penalty, the effect of locking on the performance of the biasing circuit itself, e.g., accuracy, temperature stability, bandwidth, input/output compliance voltage, and input/output resistance, has not been adequately addressed so far. A small performance degradation is reported in [8] when applying transistor width obfuscation in a PLL. Certainly, memristor-based [11] and neural network-based [12] biasing can be problematic in this regard. Calibration locking and *MixLock* do not incur any performance penalty.

The design complexity of the locking mechanism is considered to be acceptable by the designer for all techniques apart from memristor-based [11] and neural network-based [12] biasing locking. The calibration locking technique in [14] does not require any extra design effort. Logic locking is automated, thus M-S locking, i.e., [13] and *MixLock*, is automated.

As explained in Section II-B, there are specific attacks developed for biasing locking. For M-S locking, i.e., [13] and *MixLock*, we assume that state-of-the-art logic locking is used that is resilient to all known counter-attacks on logic locking. We assume also that the locked digital sections cannot be easily re-designed by an attacker, thus *MixLock* thwarts cloning. While *MixLock* obfuscates the actual core of the circuit, calibration locking [13] leaves the core circuit unprotected but obfuscates its tuning capability instead. We believe that the latter technique is sufficient to thwart cloning and not just overproduction as motivated in [13] since without tuning capability the circuit has a degraded performance trade-off, which renders it unusable. The calibration locking technique based on limiting the tuning range of AFGTs [15] is vulnerable to a lock removal attack. Thus, this technique cannot serve as a strong countermeasure against cloning.

The compound locking technique proposed in [16] employs *MixLock* for the digital section and biasing locking for the analog section. It is generally applicable to any M-S IC. However, using two locking schemes increases the area and power overhead. It inherits the security level offered by *MixLock* and, interestingly, it can resist a biasing locking attack if the two locks share key-bits. The reason is that a biasing locking attack may extract a valid key that correctly sets the bias point, yet this valid key is likely not the same as the actual secret key. Thus, applying the extracted shared key-bits of the valid key for the analog section to the locked digital section is unlikely to correctly unlock the digital section and will mislead any logic locking attack. In short, from a circuit locked via the compound technique in [16] a functional analog section can still be recovered, however unlocking the design as a whole becomes more challenging.

VII. Conclusions

We proposed *MixLock*, a generic, digitally-assisted security method for M-S ICs protecting against piracy. *MixLock* leverages logic locking of digital blocks in the signal path to lock M-S performances at system-level. The *MixLock* protection is resilient against all known attacks in both the analog and digital domain. It only acts on the digital blocks leaving analog

blocks intact. This has important implications, i.e., the analog design flow is left unchanged, the locking method is automated and applied only once the design is finalized without requiring any analog design re-iteration, the locking incurs no M-S performance penalty, and area and power overheads due to locking are low and justifiable. These properties are key for the wide adoption of *MixLock* by analog designers. In our current *MixLock* implementation we use the state-of-the-art DisORC with TRLL logic locking method. We demonstrated *MixLock* on a $\Sigma\Delta$ ADC using hardware measurements and an audio demonstrator.

REFERENCES

[1] B. Lippmann, M. Werner, N. Unverricht, A. Singla, P. Egger, A. Dübotzky, H. Gieser, M. Rasche, O. Kellermann, and H. Graeb, "Integrated flow for reverse engineering of nanoscale technologies," in *Proc. Asia and South Pacific Design Automation Conference*, 2019, p. 82–89.

[2] U. Guin, K. Huang, D. DiMase, J. M. Carulli, M. Tehranipoor, and Y. Makris, "Counterfeit integrated circuits: A rising threat in the global semiconductor supply chain," *Proceedings of the IEEE*, vol. 102, no. 8, pp. 1207–1228, 2014.

[3] A. Sanabria-Borbón, N. G. Jayasankaran, J. Hu, J. Rajendran, and E. Sánchez-Sinencio, "Analog/RF IP protection: Attack models, defense techniques, and challenges," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 68, no. 1, pp. 36–41, 2021.

[4] A. Chakraborty, N. G. Jayasankaran, Y. Liu, J. Rajendran, O. Sinanoglu, A. Srivastava, Y. Xie, M. Yasin, and M. Zuzak, "Keynote: A disquisition on logic locking," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 39, no. 10, pp. 1952–1972, 2020.

[5] N. Limaye, E. Kalligeros, N. Karousos, I. G. Karybali, and O. Sinanoglu, "Thwarting all logic locking attacks: Dishonest oracle with truly random logic locking," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 40, no. 9, pp. 1740–1753, 2021.

[6] J. Leonhard, M. Yasin, S. Turk, M. Nabeel, M.-M. Louërat, R. Chotin-Avot, H. Aboushady, O. Sinanoglu, and H.-G. Stratigopoulos, "MixLock: Securing mixed-signal circuits via logic locking," in *Proc. Design, Automation & Test in Europe Conference*, 2019.

[7] M. Yasin, A. Sengupta, M. T. Nabeel, M. Ashraf, J. Rajendran, and O. Sinanoglu, "Provably-secure logic locking: From theory to practice," in *Proc. ACM SIGSAC Conference on Computer and Communications Security*, 2017, pp. 1601–1618.

[8] V. V. Rao and I. Savidis, "Protecting analog circuits with parameter biasing obfuscation," in *Proc. IEEE Latin American Test Symposium*, 2017.

[9] J. Wang, C. Shi, A. Sanabria-Borbon, E. Sánchez-Sinencio, and J. Hu, "Thwarting analog IC piracy via combinational locking," in *Proc. IEEE International Test Conference*, 2017.

[10] V. V. Rao and I. Savidis, "Mesh based obfuscation of analog circuit properties," in *IEEE International Symposium on Circuits and Systems*, 2019.

[11] D. H. K. Hoe, J. Rajendran, and R. Karri, "Towards secure analog designs: A secure sense amplifier using memristors," in *Proc. IEEE Computer Society Annual Symposium on VLSI*, 2014.

[12] G. Volanis, Y. Lu, S. Govinda, R. Nimmalapudi, A. Antonopoulos, A. Marshall, and Y. Makris, "Analog performance locking through neural network-based biasing," in *Proc. IEEE VLSI Test Symposium*, 2019.

[13] N. G. Jayasankaran, A. Sanabria Borbon, E. Sanchez Sinencio, J. Hu, and J. Rajendran, "Towards provably-secure analog and mixed-signal locking against overproduction," *IEEE Transactions on Emerging Topics in Computing*, 2020.

[14] M. Elshamy, A. Sayed, M.-M. Louërat, A. Rhouni, H. Aboushady, and H.-G. Stratigopoulos, "Securing programmable analog ICs against piracy," in *Proc. Design, Automation & Test in Europe Conference*, 2020.

[15] S. G. Rao Nimmalapudi, G. Volanis, Y. Lu, A. Antonopoulos, A. Marshall, and Y. Makris, "Range-controlled floating-gate transistors: A unified solution for unlocking and calibrating analog ICs," in *Proc. Design, Automation & Test in Europe Conference*, 2020.

[16] K. Juretus, V. Venugopal Rao, and I. Savidis, "Securing analog mixed-signal integrated circuits through shared dependencies," in *Proc. ACM Great Lakes Symposium on VLSI*, 2019.

[17] N. G. Jayasankaran, A. Sanabria-Borbón, A. Abuellil, E. Sánchez-Sinencio, J. Hu, and J. Rajendran, "Breaking analog locking techniques," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 28, no. 10, pp. 2157–2170, 2020.

[18] V. V. Rao, K. Juretus, and I. Savidis, "Security vulnerabilities of obfuscated analog circuits," in *IEEE International Symposium on Circuits and Systems*, 2020.

[19] R. Y. Acharya, S. Chowdhury, F. Ganji, and D. Forte, "Attack of the genes: Finding keys and parameters of locked analog ICs using genetic algorithm," in *IEEE International Symposium on Hardware Oriented Security and Trust*, 2020, pp. 284–294.

[20] J. Leonhard, M. Elshamy, M.-M. Louërat, and H.-G. Stratigopoulos, "Breaking analog biasing locking techniques via re-synthesis," in *Proceedings of the 26th Asia and South Pacific Design Automation Conference*, 2021, p. 555–560.

[21] Y. Bi, J. S. Yuan, and Y. Jin, "Beyond the interconnections: split manufacturing in RF designs," *Electronics*, vol. 4, no. 3, pp. 541–564, 2015.

[22] A. Ash-Saki and S. Ghosh, "How multi-threshold designs can protect analog IPs," in *Proc. IEEE International Conference on Computer Design*, 2018, pp. 464–471.

[23] J. Leonhard, A. Sayed, M. Louërat, H. Aboushady, and H. Stratigopoulos, "Analog and mixed-signal IC security via sizing camouflaging," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 40, no. 5, pp. 822–835, 2021.

[24] J. A. Roy, F. Koushanfar, and I. L. Markov, "Ending piracy of integrated circuits," *IEEE Computer*, vol. 43, no. 10, pp. 30–38, 2010.

[25] J. Rajendran, H. Zhang, C. Zhang, G. S. Rose, Y. Pino, O. Sinanoglu, and R. Karri, "Fault analysis-based logic encryption," *IEEE Transactions on Computers*, vol. 64, no. 2, pp. 410–424, 2015.

[26] M. Yasin, J. J. Rajendran, O. Sinanoglu, and R. Karri, "On improving the security of logic locking," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 35, no. 9, pp. 1411–1424, 2016.

[27] P. Subramanyan, S. Ray, and S. Malik, "Evaluating the security of logic encryption algorithms," in *Proc. IEEE International Symposium on Hardware Oriented Security and Trust*, 2015.

[28] K. Juretus and I. Savidis, "Characterization of in-cone logic locking resiliency against the SAT attack," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 39, no. 8, pp. 1607–1620, 2020.

[29] M. Yasin, B. Mazumdar, J. Rajendran, and O. Sinanoglu, "SARLock: SAT attack resistant logic locking," in *IEEE International Symposium on Hardware Oriented Security and Trust*, 2016.

[30] K. Shamsi, M. Li, T. Meade, Z. Zhao, D. Z. Pan, and Y. Jin, "AppSAT: Approximately Deobfuscating Integrated Circuits," in *Proc. IEEE International Symposium on Hardware Oriented Security and Trust*, 2017, pp. 95–100.

[31] K. Shamsi, M. Li, D. Z. Pan, and Y. Jin, "KC2: Key-condition crunching for fast sequential circuit deobfuscation," in *Design, Automation & Test in Europe Conference & Exhibition*, 2019, pp. 534–539.

[32] P. Chakraborty, J. Cruz, and S. Bhunia, "SAIL: Machine learning guided structural analysis attack on hardware obfuscation," in *IEEE Asian Hardware Oriented Security and Trust Symposium*, 2018, pp. 56–61.

[33] D. Sisejkovic, F. Merchant, L. M. Reimann, H. Srivastava, A. Hallawa, and R. Leupers, "Challenging the security of logic locking schemes in the era of deep learning: A neuroevolutionary approach," *arXiv preprint arXiv:2011.10389*, 2020.

[34] L. Li and A. Orailoglu, "Piercing logic locking keys through redundancy identification," in *Design, Automation & Test in Europe Conference & Exhibition*, 2019, pp. 540–545.

[35] K. Juretus and I. Savidis, "Increased output corruption and structural attack resilience for SAT attack secure logic locking," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 40, no. 1, pp. 38–51, 2021.

[36] K. Shamsi, D. Z. Pan, and Y. Jin, "On the impossibility of approximation-resilient circuit locking," in *IEEE International Symposium on Hardware Oriented Security and Trust*, 2019, pp. 161–170.

[37] A. Sayed, T. Badran, M. Louërat, and H. Aboushady, "A 1.5-to-3.0GHz tunable RF sigma-delta ADC with a fixed set of coefficients and a programmable loop delay," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 67, no. 9, pp. 1559–1563, 2020.

[38] J. Leonhard, M.-M. Louërat, H. Aboushady, O. Sinanoglu, and H.-G. Stratigopoulos, "Mixed-signal hardware security using MixLock: Demonstration in an audio application," in *International Conference on Synthesis, Modeling, Analysis and Simulation Methods and Applications to Circuit Design*, 2019.

Julian Leonhard received the B.Sc. and M.Sc. in electrical and computer engineering from the Technical University of Munich, Germany, in 2013 and 2016, respectively. He wrote his B.Sc. thesis with Intel Mobile Communications, Germany, and his M.Sc. Thesis with Infineon Technologies, Germany. He obtained his PhD at Sorbonne Université, Paris, France, in 2021. Currently he is a post-doctoral researcher at the LIP6 Laboratory of Sorbonne Université. His research topics include hardware security and design-for-trust for analog and mixed-signal circuits.

Nimisha Limaye (Graduate Student Member, IEEE) is a Ph.D. candidate at the Department of Electrical and Computer Engineering at New York University, USA, and is also a Global Ph.D. Fellow with New York University Abu Dhabi, UAE. Her research interests include hardware security with particular focus on logic locking and scan locking. She received the B.E.in Electronics and Telecommunications Engineering from University of Mumbai, India, in 2015 and the M.S.in Computer Engineering from New York University, New York, USA, in 2017.

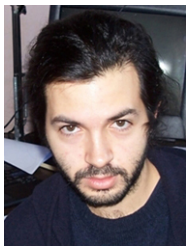
Shadi Turk received the B.Sc. degree in communication systems engineering from the Faculty of Engineering of Ain Shams University, Cairo, Egypt, in 2016 and the M.Sc. degree in electronic and computer systems from the Faculty of Science of Sorbonne University, Paris, France, in 2017. He currently works as a digital circuit design engineer at Seamless Waves, Paris, France, where he performs hardware development of high-performance wireless transceivers. He started exploring hardware security for mixed-signal circuits when he joined the Computer Science Laboratory (LIP6) of Sorbonne University, Paris, France, as a project engineer from October 2017 to February 2018.

Alhassan Sayed received the B.Sc. and the M.Sc. degrees in Electrical Engineering, from the Electronics and Communications Department, Minia University, Minia, Egypt, in 2007 and 2010, respectively. He obtained his Ph.D. degree in Electrical Engineering and Computer Science from Sorbonne University, Paris, France, in 2016. He also spent 2 years (2017-2019) in a postdoctoral research position at the same University. Dr. Sayed is currently an Assistant Professor at Minia University, Minia, Egypt. His research interests include Sigma-Delta modulation, analog and RF circuit design, Analog-to-Digital conversion, and low noise amplifiers.

Alán Rodrigo Díaz Rizo is a Ph.D. candidate at the Computer Science Laboratory (LIP6) of Sorbonne University's Faculty of Science and Engineering, Paris, France. His research interests include hardware security, cognitive radio, and radio signal processing. He received the B.Sc. in Electronics and Communication Engineering from Guadalajara University, Guadalajara, Mexico, in 2015, and the M.Sc. in Electrical Engineering from the Center for Research and Advanced Studies of the National Polytechnic Institute (Cinvestav), Mexico, in 2018.

Hassan Aboushady (Senior Member, IEEE) received the B.Sc. degree in Electrical Engineering from Cairo University, Egypt, in 1993, the M.Sc. and Ph.D. degrees in Electrical Engineering and Computer Science from Sorbonne University, Paris, France, in 1996 and 2002 respectively. Dr. Aboushady is currently an Associate Professor at Sorbonne University. His research interests include Sigma-Delta modulation, Analog/RF circuit design, Analog-to-Digital and Digital-to-Analog conversion, as well as Security in analog and mixed-signal circuits. He is the author and co-author of more than 70 publications in these areas. He is the recipient of the 2004 best paper award in the IEEE Design Automation and Test in Europe Conference, as well as the recipient and the co-recipient of the 2nd and the 3rd best student paper awards of the IEEE Midwest Symposium on Circuits and Systems in 2000 and 2003 respectively. Dr. Aboushady is an IEEE-CAS distinguished lecturer and a member of the IEEE Circuits and Systems for Communications Committee (CASCOM). He also served as an Associate Editor of the IEEE Transactions on Circuits and Systems II: Express Briefs.

Ozgur Sinanoglu (Senior Member, IEEE) is a professor of electrical and computer engineering at New York University Abu Dhabi. He obtained his PhD in Computer Science and Engineering from University of California San Diego. He has industry experience at TI, IBM and Qualcomm, and has been with NYU Abu Dhabi since 2010. During his PhD, he won the IBM PhD fellowship award twice. He is also the recipient of the best paper awards at IEEE VLSI Test Symposium 2011 and ACM Conference on Computer and Communication Security 2013. Prof. Sinanoglu's research interests include design-for-test, design-for-security and design-for-trust for VLSI circuits, where he has more than 200 conference and journal papers, and 20 issued and pending US Patents. Prof. Sinanoglu is the director of the Center for Cybersecurity at NYU Abu Dhabi. His recent research in hardware security and trust is being funded by US National Science Foundation, US Department of Defense, Semiconductor Research Corporation, Intel Corp and Mubadala Technology.



Haralampos-G. Stratigopoulos (Member, IEEE) received the Diploma in electrical and computer engineering from the National Technical University of Athens, Athens, Greece, in 2001 and the Ph.D. in electrical engineering from Yale University, New Haven, USA, in 2006. From October 2007 to May 2015 he was a Researcher with the French National Center for Scientific Research (CNRS) at TIMA Laboratory, Université Grenoble Alpes, Grenoble, France. Currently he is a Research Director with the CNRS at LIP6 Laboratory, Sorbonne Université, Paris, France. His main research interests are in the areas of design-for-test for analog, mixed-signal, RF circuits and systems, machine learning, hardware security, and neuromorphic computing. He was the General Chair of the 2015 IEEE International Mixed-Signal Testing Workshop (IMSTW) and the Program Chair of the 2017 IEEE European Test Symposium (ETS). He has served on the Technical Program Committees of Design, Automation, and Test in Europe Conference (DATE), Design Automation Conference (DAC), IEEE International Conference on Computer-Aided Design (ICCAD), IEEE European Test Symposium (ETS), IEEE International Test Conference (ITC), IEEE VLSI Test Symposium (VTS), and several others international conferences. He has served as an Associate Editor of IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on Circuits and Systems I: Regular Papers, IEEE Design & Test, and Springer Journal of Electronic Testing: Theory & Applications. He received the Best Paper Award in the 2009, 2012, and 2015 IEEE European Test Symposium (ETS).