SyncLock: RF Transceiver Security Using Synchronization Locking

Alán Rodrigo Díaz Rizo, Hassan Aboushady, Haralampos-G. Stratigopoulos
Sorbonne Université, CNRS, LIP6, Paris, France

***Abstract*—We present an anti-piracy locking-based design methodology for RF transceivers, called *SyncLock*. *SyncLock* acts on the synchronization of the transmitter with the receiver. If a key other than the secret one is applied the synchronization and, thereby, the communication fails. *SyncLock* is implemented using a novel locking concept. A hard-coded error is hidden into the design while the unlocking, i.e., the error correction, takes place at another part of the design upon application of the secret key. *SyncLock* presents several advantages. It is generally applicable, incorrect keys result in denial-of-service, it incurs no performance penalty and minimum overheads, and it offers maximum security thwarting all known counter-attacks. We demonstrate *SyncLock* with hardware measurements.**

I. Introduction

Piracy of Intellectual Property (IP) blocks in Integrated Circuits (ICs) and Systems-on-Chip (SoC) or of the entire IC or SoC arises as a major hardware security and trust threat due to the globalization of design and manufacturing tasks and their outsourcing to potentially untrusted third parties, as well as due to the increased capabilities for performing reverse-engineering of chips. Piracy includes cloning, overbuilding, remarking, and recycling. Cloning refers to illegally copying a design and reusing without the consent or knowledge of the design owner. It can be performed by rogue agents in IC/SoC integration houses and foundries or by an end-user through reverse-engineering of a legally purchased chip. Overbuilding can be performed by a foundry that holds the blueprint of the design and refers to producing and selling chips beyond the number agreed on in the contract with the chip design owner. Remarking can be performed by a test facility and refers to relabelling failing chips as functional. Recycling refers to scrapping a likely aged chip from a used board and re-entering it into the market. Piracy leads to counterfeit chips that are a serious threat for design houses (e.g., loss of know-how, sales, and brand name), governments (e.g., national security threat if counterfeit chips are used in critical infrastructure or defense), and the society as whole (e.g., counterfeits are likely to be of lower quality and have shorter lifespan).

IP/IC locking is considered as the strongest counter-measure against piracy protecting an IP/IC against potential attackers located anywhere in the supply chain, as well as against malicious end-users and recycling facilities [1]. IP/IC locking is performed by the designer and consists in embedding a lock mechanism inside the IP/IC. The lock mechanism is a circuit that is controlled by a key which is typically in the form of a digital bit-string. The lock mechanism is transparent to the IP/IC such that upon application of the correct key the nominal functionality is restored. However, applying an incorrect key corrupts the functionality. The correct key is a secret of the designer and is not shared with any potentially untrusted party, i.e, integration house, foundry, or end-user. The chip is securely activated remotely after fabrication by storing the secret key in a Tamper-Proof Memory (TPM) such that it is erased on detecting a probing attempt.

Existing techniques for locking Analog and Mixed-Signal (AMS) ICs include biasing locking [2]–[4], *MixLock* [5], and calibration locking [6]–[8]. Biasing locking aims at controlling the bias generation with the key. It may result in imprecise or unstable biasing and, besides, recently counter-attacks were proposed that break this type of defense [9]–[11]. *MixLock* leverages locking techniques in the digital domain (a.k.a logic locking) to lock an AMS IC via locking its digital section. It incurs a justifiable yet non-negligible area and power overhead. Calibration locking makes the compensation of process variations or adaptation to different operation modes key-dependent. To be secure calibration locking requires that the calibration algorithm is complex enough to be devised or re-designed in hardware by the attacker, an assumption that is not always met.

In this work, we propose a locking-based system-level security mechanism for preventing piracy of RF transceivers, called Synchronization Locking (or *SyncLock*). For invalid keys *SyncLock* corrupts the synchronization process between the transmitter and the receiver. By blocking the synchronization, wireless receivers are unable to find the start of the received frame, thus the wireless communication fails. *SyncLock* is demonstrated in hardware using the Software Defined Radio (SDR) bladeRF board from Nuand.

The rest of the paper is structured as follows. In Section II, we present *SyncLock*, including its principle of operation, locking mechanism, practicality, overheads, and security analysis. In Section III, we present the hardware platform. In Section IV, we demonstrate the locking efficiency of *SyncLock* with hardware measurements. Section V concludes the paper.

II. SyncLock

A. Principle of operation

A simplified architecture of a wireless IC with *SyncLock* embedded is shown in Fig. 1. *SyncLock* acts on two different parts of the design. First, it modifies the frame generation block at the end of the baseband digital signal processing (DSP) chain of the transmitter by corrupting the preamble of the frame. The introduced error is hard-coded such that after synthesis of the DSP it is impossible to be traced and recovered by netlist analysis. Then, it modifies the preamble generation block at the beginning of the baseband DSP chain such that the output preamble is key-controlled. To enable the

978-3-9819263-6-1/DATE22/©2022 EDAA

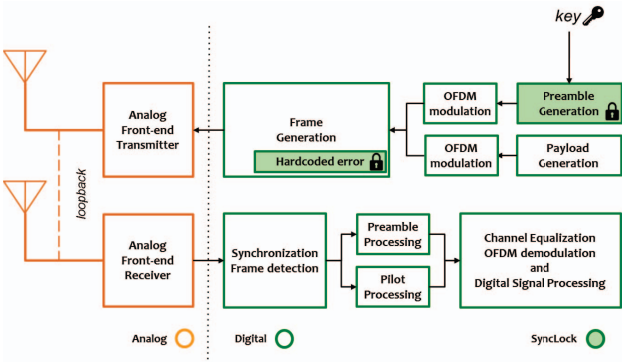

Fig. 1. Simplified architecture of a wireless device IC with *SyncLock*.

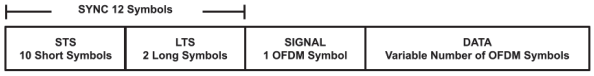

Fig. 2. PPDU frame format of an OFDM IEEE 802.11a/g transmission.

synchronization process the key must match the unknown to the attacker hard-coded error into the frame generator.

B. Preamble generation

In all wireless communication protocols, the payload is transmitted along with the physical (PHY) layer specifications. The baseband DSP prepares the payload in a frame format for transmission. The PHY Layer Protocol Data Unit (PPDU) frame format of an Orthogonal Frequency-Division Multiplexing (OFDM) IEEE 802.11a/g transmission consists of several OFDM symbols. These symbols are divided into three parts: preamble (a.k.a SYNC), header (a.k.a SIGNAL), and payload (a.k.a DATA). The preamble section is composed of two different training symbol sequences, namely a short training sequence (STS) and a long training sequence (LTS). Fig. 2 shows the PPDU of an IEEE 802.11 transmission with the above three parts as defined in the IEEE 802.11a/g standard [12]. The STS field is used for timing acquisition based on the Schmidl and Cox algorithm [13], i.e., for synchronization or frame start detection, and for rough frequency offset estimation. The LTS field is used for channel estimation and fine frequency offset estimation [12].

More specifically, as defined in the IEEE 802.11 standard [12], the nominal STS is composed of 10 repetitions of a short symbol, which is denoted herein by STS_{nom}. STS_{nom} is the sequence of 16 samples shown in Table I. Each sample is a complex number with the real (imaginary) part corresponding to the I (Q) channel, and each part has a 16-bit fixed-point representation, as shown in the second column of Table I. Therefore, STS_{nom} contains $N = (16{+}16){*}16$ bits = 512 bits. STS_{nom} is generated in the baseband DSP by the preamble generation block as shown in Fig. 1.

C. Locking mechanism

SyncLock acts specifically on the generation of STS_{nom}. The locking mechanism of *SyncLock* is divided into two parts embedded into the preamble and frame generation blocks, as

TABLE I
STS_{nom} AS DEFINED IN THE IEEE 802.11 STANDARD [12].

Sample (k)	Fixed-point (I,Q)
0	16'h02F2 , 16'h02F2
1	16'hF786 , 16'h0022
2	16'hFF23 , 16'hF2F1
3	16'h0923 , 16'hF731
4	16'h05E3 , 16'h0000
5	16'h0923 , 16'hF731
6	16'hFF23 , 16'hF2F1
7	16'hF786 , 16'h0022
8	16'h02F2 , 16'h02F2
9	16'h0022 , 16'hF786
10	16'hF2F1 , 16'hFF23
11	16'hF731 , 16'h0923
12	16'h0000 , 16'h05E3
13	16'hF731 , 16'h0923
14	16'hF2F1 , 16'hFF23
15	16'h0022 , 16'hF786

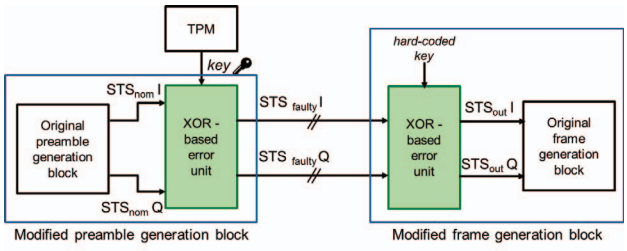

Fig. 3. Preamble and frame generation blocks modified for *SyncLock*.

shown in Fig. 3. In the frame generation block, the STS_{nom} originally generated by the preamble generation block is embedded into the frame for transmission. The design owner deliberately corrupts STS_{nom} before the creation of the frame by XORing it with a hard-coded key, denoted by key_{h-c}. In the preamble generation block, the same XOR operation is performed at the output between the key and STS_{nom}, corrupting the STS_{nom} to a faulty value, denoted by STS_{faulty}. This time, however, the TPM drives the key inputs. The equation describing the final value is $STS_{out} = (STS_{nom} \oplus key) \oplus key_{h-c}$, thus $STS_{out} = STS_{nom}$ only if $key = key_{h-c}$. Essentially, the generated STS_{nom} by the original preamble generation block is corrupted so as to neutralize the second corruption that comes down in the DSP chain at the frame generation block. The secret key must be loaded in the TPM of the chip to restore the synchronization. Applying incorrect keys introduces two uncorrelated STS corruptions at two distinct blocks of the DSP chain which breaks the synchronization.

D. Practicality

Since a synchronization process is present and necessary in any wireless communication protocol, *SyncLock* is applicable to any of them. Furthermore, since *SyncLock* only acts on the preamble generation, it is independent of the modulation scheme that is applied on the payload and, thereby, it is generally applicable for any modulation scheme. Finally, since *SyncLock* modifies only the DSP, it is independent of the Analog Front-End (AFE) architecture.

E. Overheads

1) Area Overhead: The area overhead of *SyncLock* corresponding to the two added XOR modules is computed by using

as baseline unlocked implementation an open-source IEEE 802.11 compatible SDR VHDL modem [14]. The project is called *bladeRF-wiphy* as it implements the PHY layer of the IEEE 802.11 standard in the bladeRF board. More details about the bladeRF board will be given in Section III. Starting from the unlocked implementation, we added the *SyncLock* locking mechanism into the PHY layer of the modem. For a maximum key size of 512 bits, *SyncLock* results in 1.1152% area overhead for the baseband DSP section, which when projected to the entire RF transceiver is even smaller as the area is dominated by the AFE. The overhead of the TPM is excluded as it is considered fixed for any locking mechanism.

2) Power overhead: Embedding *SyncLock* in the *bladeRF-wiphy* PHY layer implementation as above resulted in no noticeable power overhead.

3) Performance penalty: Since the *SyncLock* mechanism is entirely implemented into the baseband DSP of the RF transceiver it does not incur any performance penalty.

4) AMS IC design flow: The AFE is left intact, thus there is no change in the AMS IC design flow.

F. Security analysis

1) Threat model: We consider the most demanding threat model for a defender. We assume that the attacker is in possession of the netlist and an oracle, i.e., a working chip with the correct key applied into the TPM.

2) Brute-force and optimization attacks: The attacker searches in the key space either randomly in a brute-force manner or more efficiently by employing an optimization algorithm hoping to find a key that enables synchronization. The search is performed by simulating the design at netlist-level where the TPM is circumvented and the key inputs are and accessed directly. Resilience against this attack is achieved since: (a) the key space size is huge; (b) a single key enables synchronization, thus the optimization function behaves like a delta function on the secret key and an optimization algorithm will "zig-zag" endlessly; (c) a single simulation at netlist-level can be very time-consuming, thus the attacker in practice can perform a very limited number of trials.

3) Attacks in the digital domain: Several attacks have been developed for breaking logic locking techniques [1]. For example, attacks based on Boolean satisfiability (SAT) have shown to be very powerful recovering the key with little effort. The SAT attack computes distinguishing input patterns (DIPs) defined as inputs which produce different output for at least two different keys, and rules out incorrect keys iteratively using DIPs and querying the oracle. It does not apply to *SyncLock* since the inputs to the preamble generation block are fixed and hard-coded.

4) Removal attack: The attacker can trace the key-bits and identify and remove the XOR module in the preamble generation block. In this case, the design will be left with a hard-coded error impeding synchronization. After DSP synthesis, the small XOR module into the frame generation block that hard-codes the error is immersed in the original design and the two become inseparable. An attacker who has the non-annotated netlist in possession cannot locate this XOR module and, thereby, cannot remove it.

5) Known-plaintext attack (KPA): The KPA attack applies to stream ciphers with symmetric encryption, i.e., when the encryption and decryption processes are performed using the same encryption key. In our context, the plaintext is STS_{nom} and is known to the attacker since it is published in the IEEE 802.11 standard [12]. The attacker can apply a trial key, denoted by key_{trial}, and using $STS_{out} = (STS_{nom} \oplus key_{trial}) \oplus key_{h-c}$ and the associative and self-inverse properties of the XOR function, can recover the hard-code key as $key_{h-c} = key_{trial} \oplus STS_{nom} \oplus STS_{out}$. In the oracle chip, the attacker cannot re-write the TPM to apply a trial key, thus STS_{out} has to be extracted in simulation. The attacker can simulate a transmission and try to extract STS_{out} from the transmitted frame. A loopback connection to the receiver can be used to analyse the transmitted signal in the baseband. The attacker should be able to manually locate the STS_{out} bits. But even in this case, some STS_{out} bits will be corrupted due to quantization noise and non-linearities introduced along the signal processing at transistor-level. Thus, the transmitted STS_{out} cannot be correctly extracted and the computed key_{h-c} will be incorrect. The only approach that could work is to locate the STS_{out} signal in the transmitter's baseband, but this is challenging since the netlist is not annotated.

III. Hardware Platform

SyncLock is demonstrated in hardware using the SDR bladeRF board from Nuand. This board contains three main chips: an RF transceiver LMS6002 from Lime Microsystems, a Field-Programmable Gate Array (FPGA) Cyclone IV from Intel (formerly ALTERA), and a USB 3.0 peripheral controller FX3 from Cypress.

The complete architecture shown in Fig. 1 has been implemented in the bladeRF board. The RF transceiver LMS6002 implementing the AFE has a conventional Zero-IF architecture for both the receiver and the transmitter. It has an on-chip loopback mode, as shown in Fig. 1, thus allowing to perform Bit Error Rate (BER) measurements and check the synchronization using the same board. We assume an Additive White Gaussian Model (AWGN) channel model. The baseband DSP is available in VHDL and is implemented in the FPGA of the board. The VHDL code of the preamble and frame generation blocks is modified to insert the *SyncLock* locking mechanism and is re-embedded into the same FPGA project. The payload of the transmitted signal is modulated using Binary Phase-Shift Keying (BPSK) and is encoded into OFDM symbols.

IV. Experimental Results

Fig. 4 shows the BER of an OFDM-BPSK transmission without *SyncLock*, with *SyncLock* when applying the correct key, and with *SyncLock* when applying a randomly chosen incorrect key. A first observation is that when applying the correct key there is no BER penalty. The curves of BER without and with *SyncLock* are identical for all SNR values. The reason is that *SyncLock* modifies only the DSP and not the AFE. A second observation is that with the incorrect key the system

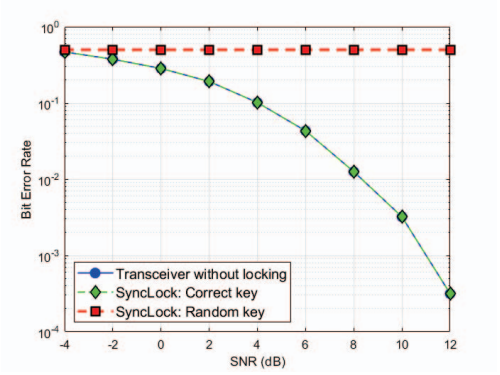

Fig. 4. Bit error rate with *SyncLock*.

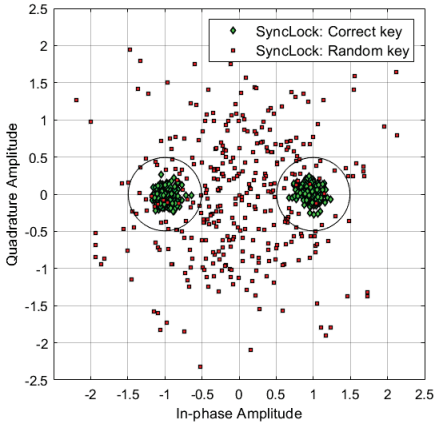

Fig. 5. Constellation diagram of the received payload.

does not synchronize and erroneously demodulates the received signal. As a result, the BER is maximum and constant across all SNR values.

Fig. 5 shows the constellation diagram of the received payload when applying the correct key and when applying a randomly chosen incorrect key. The black thin circles show the reference constellation points for a BPSK modulation. While the received signal lies inside the reference constellation using the correct key, the non-synchronized signal is randomly distributed.

Using key sampling we found that for a random initial secret key of equal size with STS_{nom}, i.e., 512 bits, all incorrect keys with Hamming Distance (HD) larger than 13 from the correct key result in no synchronization. The approximate number of incorrect keys that allow synchronization is 10^{22}, corresponding to a tiny fraction of $(10^{22}/2^{512}) \cdot 100 = 10^{-131}\%$ of the key space. Furthermore, there are 320 bit positions in the secret key which when are individually flipped result in incorrect keys with HD=1 that establish synchronization. Excluding these bit positions that correspond to the least significant bits (LSBs) of STS_{nom}, we reduce the secret key size to 192 bits while any incorrect key impedes synchronization. In this way, we can reduce the TPM size, while ensuring that all key-bits are effective and there is a single key unlocking the synchronization.

V. Conclusions

We presented *SyncLock*, a system-level locking methodology for RF transceivers generally applicable for any architecture, communication protocol, and modulation scheme. The lock is inserted into the DSP and its purpose is to break synchronization. We demonstrated that *SyncLock* offers an effective 192-bit security level and that there is a single correct key that can restore synchronization. *SyncLock* insertion is straightforward involving two XOR operations. No changes in the AFE design or the AMS IC design flow need to be made. There is no performance penalty and area overhead is around 1% of the DSP area. The *SyncLock* defense is shown also to be resilient against any known counter-attack.

Acknowledgments

This work was supported by the ANR STEALTH project under Grant N° ANR-17-CE24-0022-01. The work of A. R. Díaz Rizo was supported by the Mexican National Council for Science and Technology (CONACYT) through Fellowship.

References

[1] A. Chakraborty *et al.*, "Keynote: A disquisition on logic locking," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 39, no. 10, pp. 1952–1972, Oct. 2020.

[2] V. Rao and I. Savidis, "Performance and security analysis of parameter-obfuscated analog circuits," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, Oct. 2021, early access.

[3] J. Wang, C. Shi, A. Sanabria-Borbon, E. Sánchez-Sinencio, and J. Hu, "Thwarting analog IC piracy via combinational locking," in *Proc. IEEE Int. Test Conf. (ITC)*, Oct. 2017.

[4] 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 Symp. (VTS)*, Apr. 2019.

[5] J. Leonhard *et al.*, "Digitally-assisted mixed-signal circuit security," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, 2021, early access.

[6] N. G. Jayasankaran, A. S. Borbon, E. Sanchez-Sinencio, J. Hu, and J. Rajendran, "Towards provably-secure analog and mixed-signal locking against overproduction," in *Proc. 18th Int. Conf. Comput.-Aided Design (ICCAD)*, Nov. 2018.

[7] M. Elshamy, A. Sayed, M.-M. Louërat, A. Rhouni, H. Aboushady, and H.-G. Stratigopoulos, "Securing programmable analog ICs against piracy," in *Proc. Design, Automat. Test Eur. Conf. Exhib. (DATE)*, Mar. 2020, pp. 61–66.

[8] 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, Automat. Test Eur. Conf. Exhib. (DATE)*, Mar 2020.

[9] N. G. Jayasankaran, A. Sanabria-Borbón, A. Abuellil, E. Sánchez-Sinencio, J. Hu, and J. Rajendran, "Breaking analog locking techniques," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, Oct. 2020.

[10] 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 *Proc. IEEE Int. Symp. Hardw. Oriented Secur. Trust (HOST)*, Dec. 2020, pp. 284–294.

[11] J. Leonhard, M. Elshamy, M.-M. Louërat, and H.-G. Stratigopoulos, "Breaking analog biasing locking techniques via re-synthesis," in *Proc. 26th Asia South Pacific Design Automat. Conf.*, Jan. 2021, p. 555–560.

[12] IEEE, "IEEE standard for information technology—telecommunications and information exchange between systems local and metropolitan area networks—specific requirements - part 11: Wireless LAN medium access control (MAC) and physical layer (PHY) specifications," *IEEE Std 802.11-2016 (Revision of IEEE Std 802.11-2012)*, pp. 1–3534, 2016.

[13] T. M. Schmidl and D. C. Cox, "Robust frequency and timing synchronization for OFDM," *IEEE Trans. Commun.*, vol. 45, no. 12, pp. 1613–1621, Dec. 1997.

[14] Nuand, "Open-source ieee 802.11 compatible software defined radio vhdl modem (bladeRF-wiphy)," https://github.com/Nuand/bladeRF-wiphy/, Online.