Run-time integrity monitoring of untrustworthy analog front-ends

Heba Salem* and Nigel Topham†
School of Informatics, The University of Edinburgh, Edinburgh, United Kingdom
Email: *s1573838@ed.ac.uk, †nigel.topham@ed.ac.uk

***Abstract*—Recent advances in hardware attacks, such as cross talk and covert channel based attacks, expose the structural and operational vulnerability of analog and mixed-signal circuit elements to the introduction of malicious and untrustworthy behaviour at run-time, potentially leading to adverse physical, personal, and environmental consequences.**

One untrustworthy behaviour of concern, is the introduction of abnormal/unexpected frequencies to the signals at the analog/ digital interface of a SoC, realised through intermittent bit-flipping or stuck-at-faults in the middle and lower bits of these signals. In this paper, we study the impact of these actions and propose integrity monitoring of signals of concern based on analysing the temporal and arithmetic relations between their samples. This paper presents a hybrid software/ hardware machine-learning based framework that consists of two phases; a run-time monitoring phase, and a trustworthiness assessment phase. The framework is evaluated with three different applications and its effectiveness in detecting the untrustworthy behaviour of concern is verified. This framework is device, application, and architecture agnostic, and relies only on analysing the output of the analog front-end, allowing its implementation in SoCs with on-chip and custom analog front-ends as well as those with outsourced and commercial off-the-shelf (COTS) analog front-ends.

***Index Terms*—Hardware security, Hardware trust, Run-time monitoring**

I. Introduction

Analog front-ends are sets of analog circuits designed to condition real-world analog signals and prepare them for analog to digital conversion and later digital processing. Analog front-ends consist of sensors, filters, amplifiers, and potentially application-specific circuitry. The output of the analog front-end is fed to analog to digital converters (ADCs), which are mixed-signal circuits that transform the conditioned analog signals into a form that is comprehensible to the digital processing, control, and decision making parts of a given IC. Ensuring the operational security of the analog front-end and the ADC is paramount, as a corrupted analog front-end and/or ADC implies a corrupted system, regardless of the safety and security measures implemented in the digital domain. The trust and security of analog front-ends is threatened by a range of malicious acts including counterfeiting, hardware Trojan (HT) insertion, and frequency injection attacks [1] [2] [3].

This paper addresses the problem of detecting low-level malicious actions or untrustworthy behaviour (UB) occurring from within an ADC or the upstream sensors, amplifiers, and filters that provide the ADC with analog inputs. The malicious action of concern is output modification, in which the correctness and integrity of one of the outputs of the front-end elements is adversely affected. As the output of the ADC dictates the decisions taken downstream in the system, and is often accepted blindly as a true representation of the front-end's analog input, without continuously verifying this assumption, this issue must be addressed more thoroughly. The hardest modifications to detect are those that would be imperceptible to the observer of individual ADC outputs, but which over time have the potential to cause significant disruption in the downstream digital subsystems that rely on the ADC's output. The malicious actions may result, for example, in intermittent output bit-flips or/and stuck-at-faults.

Such actions and anomalies may originate from HTs, fault injection attacks, IP counterfeiting, and unintentional hardware faults. The non-permanent nature of faults introduced by HTs and fault injection attacks in addition to their stealth and unpredictability, and the obscured nature of defects expected of counterfeited cloned or out-of-specification ICs, all contribute to the difficulty of detecting such faults and errors or the vulnerabilities leading to them in the pre-deployment stages, and dictate the need for effective run-time monitoring techniques.

The problem: How to ensure the run-time correctness and integrity of the signals at the digital/analog interface, while accounting for the variability in the types, architectures, and applications of analog front ends and ADCs?

The proposal: In this paper, we propose a framework for machine-learning based run-time security monitoring of the signals at the digital/ analog boundary of SoCs. The proposed framework involves extracting mathematical and temporal relations between the outputs of the ADC at run-time, and projecting these relations on machine learning models that have been trained on the same mathematical and temporal relations obtained during correct and secure operation of the system. Any notable discrepancy between the projected data and the pre-trained data is an indication of incorrectness in the run-time output. By basing our monitoring technique solely on the outputs of ADCs, and considering the entire analog front-end as a black box, we are able to achieve full independence from the type and architecture of the monitored ADC and analog front-end. This is effectively a device agnostic run-time integrity monitoring of the analog front-end. This also has the additional benefit of detecting hardware faults in the analog front-end as well as malicious attacks. Identifying the source of a UB is out of scope for this paper, however, this could be performed by isolated security testing of the circuit elements of concern.

978-3-9819263-7-8/DATE23/© 2023 EDAA

II. The threat vector

Some of the most safety-critical electronic systems such as healthcare systems, avionics, manufacturing systems, and infrastructure control systems depend on analog front-ends to sample real-world signals. A successful attack on the analog front-end is a serious threat in any safety-critical system, potentially resulting in a catastrophic impact on personal safety, economic stability, and the natural environment.

In the context of HTs, the malicious actions of concern in this paper; bit-flipping and stuck-at-faults could be realized in a number of different ways from the addition of a single logic gate and connecting it to the internal ADC wires responsible for driving ADC outputs, to the utilization of capacitive cross-talk [4].Such cross-talk based HTs could be introduced by an untrusted foundry, by adding or moving wires to increase capacitive crosstalk in the proximity of wires carrying signals of interest. A capacitive crosstalk HT could force a targeted wire to stay at logic high for several cycles, leading to a sequence of erroneous outputs [4]. Pre-existing counters in some ADCs could be utilized in activating an injected HT after a certain sequence of events. Analog triggers linked to temperature or pressure and driven by on-chip sensors [5], or activated by temporal relationships measured using capacitive charging [6] [7] could also be used in activating HTs in analog and mixed signal circuits. The work in [7] presents the possibility of using the reverse saturation current of an on-chip diode and the leakage current of thin-oxide NMOS devices in charging capacitive HT triggers, delaying hence the HT activation and de-linking it from the triggering event. These advances in HT realization and activation show the viability of inserting a HT with minimal or near-zero footprint, fitting, therefore, in the small circuitry of analog and mixed signal circuits and increasing their vulnerability to HT attacks.

Analog and mixed-signal ICs and IPs, such as sensors, amplifiers, and converters are the most reported type of counterfeited semiconductor parts [1]. Likely causes are their long life cycle and small circuitry. Additionally, IP Counterfeiting is currently a prominent concern in the light of the shortage in genuine IC supply as caused by the Covid-19 pandemic and the China-US trade wars. Faulty and erroneous operation and low-integrity outputs are expected from counterfeited IPs, especially those of the recycled, defective, cloned, or out-of-specification types due to them nearing their original end-of-life, being rejected and discarded by their original manufacturers, or being based on poor and low-skilled reverse engineering and cloning. Fault injection attacks such as those based on controlled and targeted electromagnetic waves and focused laser beams are also known to cause sudden transistor switching and/ or signal flipping.

III. The proposal

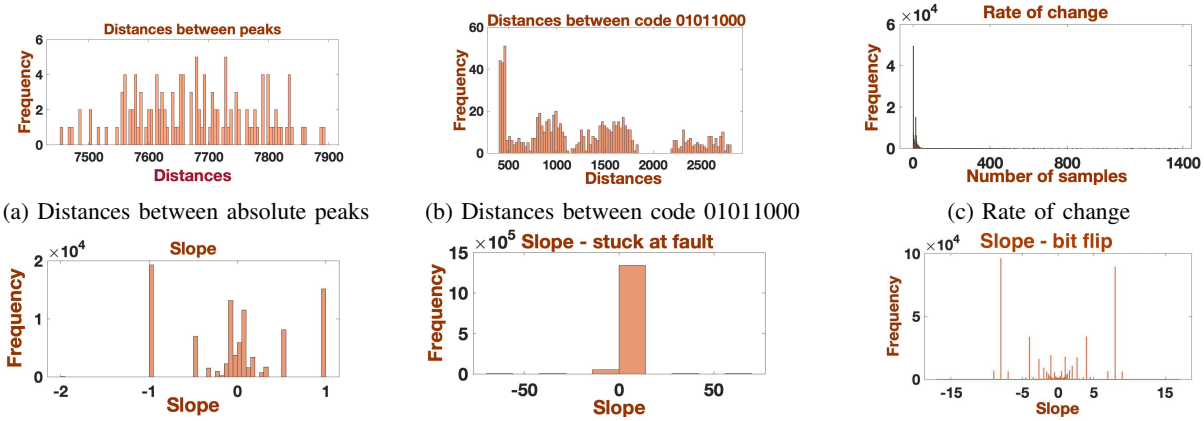
Our proposed approach to run-time integrity monitoring of the output of the analog front-end, is based on characterizing temporal and mathematical relationships between the ADC's output codes to verify if they are an actual, correct, and trustworthy representation of the analog input. The authors studied and investigated several generic relationships that are reflective of temporal and mathematical signal properties, with the aim of identifying a single relationship for systematic use in establishing references of signal integrity. One relationship is related to the location of maximum levels or peaks and the temporal distances (in number of samples) between consecutive ones. Another relationship is the distance or number of samples between certain output codes, which is generally suitable for characterizing signals of stochastic and non-uniform nature. The rate or frequency of change in the digital output code is another investigated relationship, quantified by the number of samples between changes in the code. Another relationship is the slope, or the ratio of the arithmetic change (difference in value) in the output code to the temporal change (number of samples), which effectively quantifies the frequency, factor, and direction of change in the output of an ADC.

Figure 1 shows histograms of the investigated relationships, when extracted from the digital codes of an 80bpm electrocardiogram (ECG) signal. Each of the relationships shows a distinct distribution and could be used in establishing references of signal integrity. However, distances between peaks is a relationship only applicable to periodic signals, whereas distances between specific codes is tied to the possibly fluctuating occurrences of these codes. An issue with the rate of change is that its distribution would not be altered if abnormal bit-flips coincide with actual changes in the code. To avoid these shortcomings, the slope, which reflects both the temporal and arithmetic changes in the digital output codes of ADCs, was chosen for our framework. The distribution of slope values is also noticeably sensitive to bit-flips and stuck-at-faults, as could be seen when comparing figures 1d, 1e, and 1f, which show histograms of the slopes of the 80bpm ECG signal, in the normal case, in case of a stuck-at-fault at bit 4, and in case of flipping of bit 4, respectively.

A. The framework

The proposed framework, presented in figure 2, consists of two phases; a run-time monitoring and incorrectness detection phase, and a trustworthiness assessment phase.

1) Phase 1: run-time monitoring and incorrect output detection: In order to use slopes (or any other relationship) as a reference in live detection of incorrectness in the output of the front-end (represented by the ADC output), a model based on the correct and baseline values of this relationship should be constructed and used for periodic comparison with the same relationship when obtained from live operation. Therefore, a requirement of the framework is that the type of the input signal is known, which is usually pre-defined and directly related to the application of the SoC. As slope values for a given signal will adopt a distribution of non-defined shape and with multiple peaks, they can be modelled using non-parametric density estimation machine learning models such as Kernel Density Estimation (KDE). In our approach, we train KDE models on the trusted pre-deployment slopes. Once the system is in operation, arithmetic and temporal distances between changes in the output code are collected and used in run-time slope



(a) Distances between absolute peaks (b) Distances between code 01011000 (c) Rate of change

(d) Slopes (e) Slopes when bit 4 is stuck at logic '0' (f) Slopes when flipping bit 4

Fig. 1: Histograms of the temporal and arithmetic relationships calculated from the 80bpm ECG signal

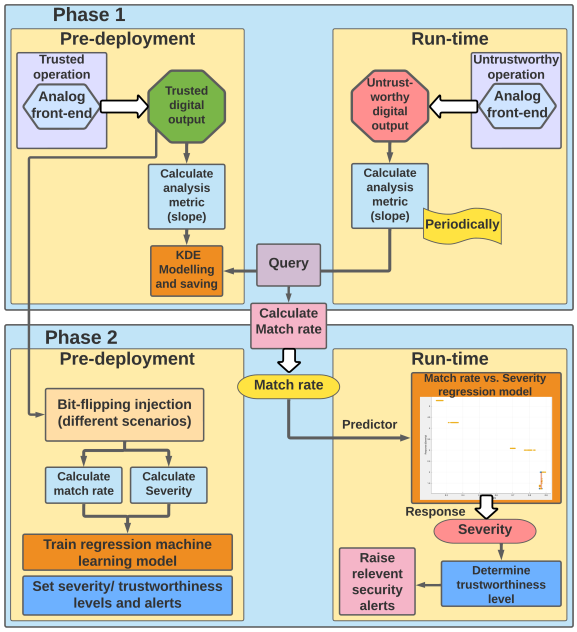


Fig. 2: The proposed analog front-end run-time integrity monitoring and trustworthiness assessment framework

calculation. These slopes are queried periodically in the pre-trained and saved KDE models. The querying process provides an indication of whether the run-time slope values have some probability in the saved KDE model, implying whether similar slopes were "seen" by the KDE model in the training stage. For a rigorous approach, the probabilities obtained from the query process may be further investigated and verified. However, for a more generic approach, we follow our query process by a match rate calculation. The match rate is the number of run-time slope values that return some probability when queried in the KDE model divided by the total number of run-time slope values. The match rate metric provides a quantified measure of how often the slopes calculated at run-time match (or are close to) those observed during trusted operation. The lower the match rate, the greater is the significance and likelihood of incorrectness.

2) Phase 2: trustworthiness assessment: The second phase of the framework is concerned with automatically and systematically distinguishing trusted from untrusted operation of the analog front end. To perform this trustworthiness assessment, we propose a severity metric that provides a quantified mapping between the calculated match rates and the severity of deviation from the correct conditioning and conversion of a given input signal. In our experiments, we emulated incorrectness in the output of the analog front-end by introducing occasional bit-flips and stuck-at-faults to the ADC's output, given that, as presented in section II, these are realistic scenarios resulting from different threat vectors such as cross-talk based HTs and electromagnetic fault injection attacks. The proposed quantification of the amount of incorrectness is, therefore, tied to the position of the bit targeted by the bit-flipping or stuck-at-fault and by the frequency of the activation/ occurrence of this flipping. The proposed severity metric is calculated through the formula presented as, $severity = (significance_{bit} * weight_{bit}) + (frequency * weight_{frequency})$. This formula is a weighted sum of the significance of the targeted bit (e.g. a significance value of 2 for attacks targeting bit 2), which is given a weight of '1', and the frequency of the bit-flipping (e.g. a value of 0.5 for bit-flipping targeting 50% of output codes over a specified period), and this frequency is assigned a weight of 0.5, reflecting our experimental observation that the significance of the targeted bit has greater impact on the distribution of slope values comparing to the impact of the frequency of activation. The mapping of the calculated severity to the targeted bit and the frequency of the bit-flipping, when applying different bit-flipping scenarios to the ECG signal, is presented in figure 3. Whereas, the mapping of the resultant match rates to the severity is presented in figure 4. In the application of the proposed framework, the SoC integrator would induce different bit-flipping scenarios to the output of the ADC in the pre-deployment stages. The resultant modified codes are then to be used in calculating the match rates (following the process presented in phase 1) and the severity of

the different bit-flipping scenarios. The severity and match rates are then used in training a regression-based machine learning model. At run-time, whenever new match rates are obtained, they are provided as predictors to this model to obtain the response, the severity of the incorrectness, a clear and direct indicator of the extent and frequency of the attack or fault. Furthermore, levels of trustworthiness are to be assigned to the different severity values with each of the trustworthiness levels mapped to certain alerts and appropriate remedial actions. Potential remedial or counter actions include offline testing of the individual elements in the front-end and filtering the front-end output to smooth out any minor incorrectness. In systems with element duplication/redundancy, the framework could also be used to signal the point of time when the secondary elements should replace the main untrustworthy ones.

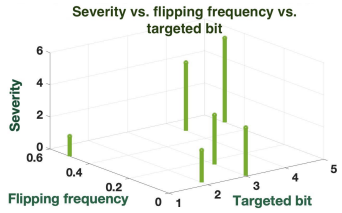


Fig. 3: Severity vs. flipping frequency vs. targeted bit

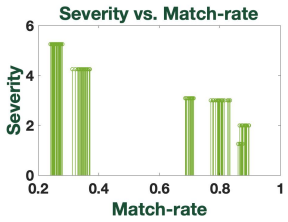


Fig. 4: Severity vs. match-rate

IV. Methodology and evaluation

A. The experimental setup

The 8-bit arithmetic tracking successive approximation register ADC simulator from [8] was used for the A/D conversion of different types of input signals pertaining to three different applications[1]; The ECG signal from the ECGSYN tool-set [9], readings of flow and level sensors from the secure water treatment system developed by iTrust in the centre for Research in Cyber Security at Singapore University of Technology and Design [10], and level and flow readings of airplane fuel tank sensors [11]. The obtained digital output in each of these cases was then analyzed and the slopes were calculated using Matlab. The KDE modelling was performed with the KDE implementation provided by the Scikit-learn tool [12], a Tophat kernel and 0.1 bandwidth were chosen so that the generated models are moderately detailed around the trained-on slope values. For the severity versus match rate machine learning, the Fine tree regression model provided by Matlab regression learner was used. For the ECG case, an RMSE of 0.11865 was obtained when training the Fine Tree regression model on the severity versus match-rate of different bit-flipping scenarios, indicating the effectiveness of this approach.

[1]The signals were modified via amplification/ attenuation and interpolation to prepare them for the A/D conversion and to obtain a large number of samples for machine learning purposes.

B. The implementation

The proposed implementation of the framework is of hybrid software/hardware nature. The KDE modelling and the regression machine learning parts are performed in software, limiting the overhead on hardware and allowing implementation in lightweight systems. Moreover, all of the steps performed in the pre-deployment stage of the second phase of the framework, could be efficiently preformed in software once digital output from trusted pre-deployment operation of the ADC is obtained.

The hardware part is concerned with the live tracking of the amount and time of change in the digital output of the ADC and is implemented independently of the analog front-end; at the analog/ digital interface of the SoC. A comparator and a substractor are used to detect changes in the ADC output and determine the amount of this change. Simultaneously, a counter counts the samples between changes in the output code. The outputs of the substractor and counter could be saved in logs for periodic calculation of slopes either in software or hardware. After querying the slopes in the saved KDE models and once the match rate is obtained, the run-time part of the second phase of the framework is also performed fully in software.

C. Use case 1: The electrocardiogram

The Electrocardiogram (ECG) signal is the recording of the electrical activity of the heart (in voltage vs. time), and is used in diagnosing many heart conditions and breathing disorders. The conditioning and A/D conversion of the ECG signal could be attacked to introduce errors and failures in the diagnostic process. One attack scenario would cause confusion in the diagnosis by adding noise and additional peaks or dips in the ECG signal, which could be easily achieved by malicious bit-flipping or stuck at faults, and hence is addressed by the proposed framework. Figure 1f shows the histograms of the slopes calculated from the bit 4-flipping affected ECG signal (UB2 in table I) and as visible the range of slope values doesn't conform with that of the correctly converted signal (figure 1d).

As the ECG signal varies naturally (within clinically identified limits) from one person to another, and to allow the KDE model to distinguish those variations from anomalies, slopes calculated from 50 different variations of the 'digitized' ECG signal were used in training the reference KDE model. The KDE model was then queried with slope values calculated from single variations of the ECG signal (reflecting the ECG of an individual) when affected by different bit-flipping scenarios. Table I shows the tested scenarios and the obtained match rates. Each of the match rates presented in table I is the average of match rates obtained from implementing the framework with different variations of the ECG signal when infected with the respective UB scenarios. When referring to the severity versus match-rate mapping shown in figure 4, UB 1 with a match rate

TABLE I: The bit-flipping scenarios introduced to the ECG signal and the obtained match rates

UB	Frequency	affected bit	match rate
UB 1	50% of code (random)	bit 5	26.7%
UB 2	50% of code (random)	bit 4	34.67%
UB 3	16% of code (random)	bit 3	70.4%
UB 4	3% of code (random)	bit 2	88.6%
UB 5	50% of code (random)	bit 1	88.79%
UB 6	0.3% of code (random)	bit 3	79.9%

of around 26.7% implies a severity greater than 5, indicating a high level of untrustworthiness, whereas, UB 5 with a match rate of around 88.79% implies a low severity level, reflecting the low significance of the affected bit.

D. Use case 2: The water treatment system

Industrial control systems (ICS) used in the vital sectors of power transmission, water treatment and distribution, and oil refining, commonly deploy sensors and ADCs for measuring and detecting the instantaneous changes and quantities of different parameters pertaining to the specific application. These measures are often used in controlling and dictating the actions performed by the rest of the system. Therefore, an attacker aiming to cause catastrophic damage and destruction in an ICS while maintaining a small attack footprint would likely target the analog front-end. One such attack was demonstrated in [2], in which ICS-level Sigma Delta ADCs were tricked to accept and convert "malicious" input frequencies potentially causing abnormal vibrations in the motor or turbine in the system, and leading to permanent damage and shut-down of the system.

The proposed framework was implemented on normal-operation sensor readings from the secure water treatment system (SWAT) developed by iTrust [10], after they were converted to digital. The tested-on sensor readings are those of the LIT101 sensor, a level sensor responsible of measuring and reporting the level of water in the raw water tank of the SWAT system, and those of the FIT301 water-flow sensor. Introducing bit-flips to these sensor readings emulates the malicious intent of causing a tank overflow/ underflow and potentially damaging the valves and pumps connected to the tank. The readings of the sensors were divided to pre-deployment trusted readings, used in KDE modelling, and run-time untrustworthy readings, injected with bit-flips. The readings from the LIT101 and the FIT301 exhibited similar match rates when affected by the same UB scenario. The introduced UB scenarios and the match rates (average) obtained are presented in table II. Figures 5a and 5b show the range of slopes obtained from the normal trusted behaviour and UB readings of the LIT101 sensor, respectively. The visible differences in the slopes, which is reflected in the calculated match rates, is a clear indicator of the effectiveness of using the proposed framework in detecting anomalies and incorrectness in the output of the analog front end in ICS.

E. Use case 3: The airplane fuel tank

Some of the most safety-critical electronic systems are avionics, given their direct and immediate impact on human lives. A vital system in airplanes is the aircraft fuel distribution system (AFDS) that is responsible of storing and distributing fuel. Typically, an AFDS consists of engines, fuel-storing tanks, valves, and jettison points. The state of fuel in the tanks is constantly monitored by level and temperature sensors, whereas the flow of the fuel in the pipes is measured by several flow meters [13]. The introduction of bit-flips to such sensors could lead to misjudgement of the amount of fuel in the tanks or its flow rate in the pipes, negatively affecting the decisions taken downstream in the system.

TABLE II: The bit-flipping scenarios introduced to the SWAT level and flow sensor readings and the obtained match rates

UB	Frequency	affected bit	match rate
UB 1	50% of code (random)	bit 5	23.09%
UB 2	50% of code (random)	bit 4	32.64%
UB 3	28% of code (random)	bit 3	63.52%
UB 4	14% of code (random)	bit 2	81.77%
UB 5	20% of code (random)	bit 1	84.96%
UB 6	1.4% of code (random)	bit 3	78.34%
UB 7	1.4% of code (random)	bit 6	38.09%

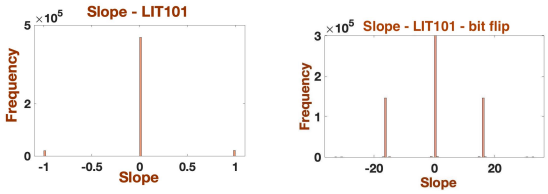
TABLE III: The bit-flipping scenarios introduced to the AFDS FTL and CLF sensor readings and the obtained match rates

UB	Frequency	affected bit	Match rate	
			FTL	CLF
UB 1	50% of code (continuous)	bit 4	22.22%	33.33%
UB 2	50% of code (continuous)	bit 5	28.57%	50%
UB 3	6% of code (random)	bit 3	18%	30%
UB 4	25% of code (continuous)	bit 6	25%	60%

Readings of fuel level sensors and fuel flow meters were obtained from the normal scenario dataset provided in [11]. The tested-on readings are those of the front and the central tank level sensors and the center to front tank flow meter, denoted as FTL, CTL, and CLF, respectively. For a more continuous representation, the readings were interpolated before converting them to digital. KDE models were then constructed from the slopes obtained from the "trustworthy" digital codes. Various bit-flipping scenarios were introduced to the readings, the proposed framework was implemented, and the match rates were calculated. The implemented UB scenarios and the resultant match rates for the sensors FTL and CLF are presented in table III, Figure 6 shows the baseline and the UB-affected (UB 4) readings of the FTL sensor and histograms of the slopes calculated from them. Although the slopes from both cases peak at the same value, the range of slopes is significantly different.

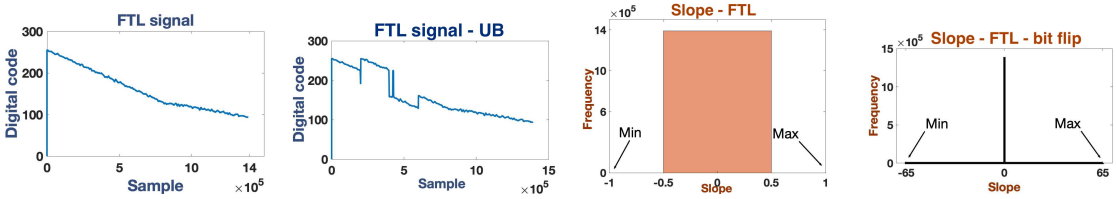
V. Related work

A small number of publications addressed a similar threat model to that addressed in this paper. In [3], the authors introduced bit monitoring based on verifying the arithmetic difference between consecutive digital codes as a countermeasure against HT-induced missing-code errors. However, the introduced bit-monitoring convention applies to certain types of signals and is more suitable for occasional verification of the correctness of the A/D conversion. In the framework presented in [14], multiple hashes are generated for internal SoC signals in multiple internal stages, and are used for verifying the integrity of the signals as they propagate in the SoC. By focusing on the security of the intermediate signals in



(a) Baseline slope histogram (b) Bit-flip slope histogram

Fig. 5: Histograms of the slopes of the baseline and the bit-flipping infected LIT101 signal



(a) Baseline FTL signal (b) Bit-flipped FTL signal (c) Baseline slope histogram (d) Bit-flip slope histogram

Fig. 6: Baseline and bit 6-flip infected FTL signals and histograms of the corresponding slopes

SoCs and implementing the security measures in independent security elements, this framework addresses a similar issue to that addressed in this paper. Nevertheless, it assumes that the outputs of the analog-front end and ADC are trusted and could be used to generate reference hashes. In a similar approach to that proposed in this paper, an approach combining statistical analysis with machine learning techniques was proposed to distinguish between trustworthy and untrustworthy operation of analog circuits in [15]. This work, however, mainly focuses on information-leaking HTs in wireless cryptography circuits.

VI. Conclusion

The work presented in this paper is, to the best of our knowledge, the first to address, at run-time, the issue of trustworthiness at the boundary between an analog front-end and the digital domain, and in a manner which is agnostic to both the device and its application. The proposed machine-learning framework provides a mechanism for characterizing the output of the analog front-end by modelling the mathematical and temporal relationships extracted from the output codes of the analog front-end. Consequently, these models serve to establish knowledge about the integrity, correctness, and trustworthiness of the output of the analog front-end at run-time. The framework is independent of the system it monitors, and its hybrid software/hardware approach enables its adoption in a wide range of applications regardless of the analog-front end, and whether it is implemented off-chip or on-chip.

References

[1] M. M. Alam, S. Chowdhury, B. Park, D. Munzer, N. Maghari, M. Tehranipoor, and D. Forte, "Challenges and Opportunities in Analog and Mixed Signal (AMS) Integrated Circuit (IC) Security," *Journal of Hardware and Systems Security*, vol. 2, no. 1, pp. 15–32, Mar. 2018. [Online]. Available: https://doi.org/10.1007/s41635-017-0024-z

[2] A. Bolshev and G. Gonzalez. (2016) How to fool an adc or how to hide the destruction of a turbine with the help of dsp. [Online]. Available: https://www.blackhat.com/docs/eu-16/materials/eu-16-Gonzalez-How-To-Fool-An-ADC-Part-II-Or-Hiding-Destruction-Of-Turbine-With-A-Little-Help-Of-Signal-Processing.pdf

[3] S. Taheri, J. Lin, and J.-S. Yuan, "Security interrogation and defense for sar analog to digital converter," *Electronics*, vol. 6, no. 2, 2017. [Online]. Available: https://www.mdpi.com/2079-9292/6/2/48

[4] C. Kison, O. M. Awad, M. Fyrbiak, and C. Paar, "Security implications of intentional capacitive crosstalk," *IEEE Transactions on Information Forensics and Security*, vol. 14, no. 12, pp. 3246–3258, 2019.

[5] S. Ghandali, D. Holcomb, and C. Paar, "Temperature-based hardware trojan for ring-oscillator-based trngs," 2019.

[6] T. Yang, A. Mittal, Y. Fei, and A. Shrivastava, "Large delay analog trojans: A silent fabrication-time attack exploiting analog modalities," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 29, no. 1, pp. 124–135, 2021.

[7] K. Yang, M. Hicks, Q. Dong, T. Austin, and D. Sylvester, "A2: Analog malicious hardware," in *2016 IEEE Symposium on Security and Privacy (SP)*, 2016, pp. 18–37.

[8] R. Inanlou, M. Safarpour, and O. Silvén, "Arithmetic tracking adaptive sar adc for signals with low-activity periods," *IEEE Access*, 2020.

[9] P. McSharry, G. Clifford, L. Tarassenko, and L. Smith, "A dynamical model for generating synthetic electrocardiogram signals," *IEEE Transactions on Biomedical Engineering*, vol. 50, no. 3, pp. 289–294, 2003.

[10] J. Goh, S. Adepu, K. N. Junejo, and A. Mathur, "A dataset to support research in the design of secure water treatment systems," in *Critical Information Infrastructures Security*, G. Havarneanu, R. Setola, H. Nassopoulos, and S. Wolthusen, Eds. Cham: Springer International Publishing, 2017, pp. 88–99.

[11] Y. Gheraibia, S. Kabir, K. Aslansefat, I. Sorokos, and Y. Papadopoulos, "Safety + ai: A novel approach to update safety models using artificial intelligence," *IEEE Access*, vol. 7, pp. 135 855–135 869, 2019.

[12] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[13] S. Kabir, M. Walker, and Y. Papadopoulos, "Dynamic system safety analysis in hip-hops with petri nets and bayesian networks," *Safety Science*, vol. 105, pp. 55–70, 2018. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0925753517314911

[14] T. Wehbe, V. J. Mooney, O. T. Inan, and D. C. Keezer, "Securing Medical Devices Against Hardware Trojan Attacks Through Analog-, Digital-, and Physiological-Based Signatures," *Journal of Hardware and Systems Security*, vol. 2, no. 3, pp. 251–265, Sep. 2018. [Online]. Available: https://doi.org/10.1007/s41635-018-0040-7

[15] Y. Jin, D. Maliuk, and Y. Makris, *Hardware Trojan Detection in Analog/RF Integrated Circuits*. Cham: Springer International Publishing, 2016, pp. 241–268. [Online]. Available: https://doi.org/10.1007/978-3-319-14971-4$_7$