High Speed ASIC Implementations of Leakage-Resilient Cryptography

Robert Schilling[*†], Thomas Unterluggauer[*], Stefan Mangard[*], Frank K. Gürkaynak[‡], Michael Muehlberghuber[‡], Luca Benini[‡]
[*]Graz University of Technology, Austria
firstname.lastname@iaik.tugraz.at
[†]Know-Center GmbH
[‡]Integrated Systems Laboratory, ETH Zurich, Switzerland
{kgf,mbgh,lbenini}@iis.ee.ethz.ch

***Abstract*—Embedded devices in the Internet-of-Things require encryption functionalities to secure their communication. However, side-channel attacks and in particular differential power analysis (DPA) attacks pose a serious threat to cryptographic implementations. While state-of-the-art countermeasures like masking slow down the performance and can only prevent DPA up to a certain order, leakage-resilient schemes are designed to stay secure even in the presence of side-channel leakage. Although several leakage-resilient schemes have been proposed, there are no hardware implementations to demonstrate their practicality and performance on measurable silicon.**

In this work, we present an ASIC implementation of a multi-core System-on-Chip extended with a software-programmable accelerator for leakage-resilient cryptography. The accelerator is deeply embedded in the shared memory architecture of the many-core system, supports different configurations, contains a high-throughput implementation of the 2PRG primitive based on AES-128, offers two side-channel protected rekeying functions, and is the first fabricated design of the side-channel secure authenticated encryption scheme ISAP. The accelerator reaches a maximum throughput of 7.49 Gbit/s and a best-case energy efficiency of 137 Gbit/s/W making this accelerator suitable for high-speed secure IoT applications.

***Index Terms*—ASIC, cryptography, IoT, leakage-resilience, security**

I. Introduction

Security and privacy are a central challenge for embedded devices in the domain of the Internet-of-Things (IoT). Consequently, cryptography is more and more commonly deployed in such devices. However, as IoT devices evolve in functionality, these devices increasingly work on larger amounts of data, which results in demand for high-throughput cryptography. To address the security of these high-performance IoT applications, we focus on two particular cryptographic settings. The first setting, which we call Netcom, is the communication between two parties that use a pre-shared secret key to transmit data securely by using encryption and/or authentication. This setting, e.g., occurs in embedded network devices which often have high throughput demands. The second setting, which we call BulkStore, is the encrypted and/or authentic storage of data. For encryption, and analogously for authentication, this setting means that data is encrypted and stored first and then decrypted multiple times either by the same or different parties. This setting often involves bulk data processing and, for example, occurs for the encryption of storage devices.

However, the ubiquitous physical exposure of IoT devices makes them vulnerable to side-channel attacks as well. In a side-channel attack, an attacker collects side-channel leakage, e.g., the power consumption or the EM radiation, while the device performs a cryptographic operation and uses this additional information to diminish the overall security provided by the cryptographic algorithms. For example, differential power analysis (DPA) extracts the secret key of, e.g., a block cipher, from analyzing power measurements taken during the en-/decryption of multiple different inputs. While there are mechanisms to protect cryptographic implementations against side-channel attacks, a promising, protocol-level countermeasure suitable for high-performance applications is leakage-resilient cryptography. Leakage-resilient schemes are designed to stay secure even in the presence of a certain amount of side-channel leakage and without dedicated DPA countermeasures in the implementation.

The challenge arising with IoT devices withstanding side-channel attacks is to fit the cryptographic functionality to the performance budget of the device that is severely limited by its small size and restrictions on the available power. While there are many different cryptographic ASIC implementations offering side-channel protection for the IoT [14], [13], there are hardly any hardware implementations of leakage-resilient cryptography available that suit the purpose of high-performance IoT applications. To cope with side-channel attacks, the high-throughput demands, and the constraints of IoT devices, leakage-resilient schemes for both Netcom and BulkStore also need to be implemented as efficiently as possible.

Our Contribution. Our work considers an attacker having physical access to an IoT device. Hence, the attacker can perform side-channel attacks such as measuring the power consumption. However, decapsulating the chip and active attacks such as fault attacks are out of scope. To meet the previous requirements, this work presents an ASIC implementation that offers leakage-resilient encryption for high-throughput IoT applications in both settings, NETCOM and BULKSTORE. In detail, our contributions are as follows:

- We propose a custom System-on-Chip (SoC) extended with a flexible software-programmable cryptographic accelerator supporting leakage-resilient operating modes.
- We show that leakage-resilient operating modes can be efficiently implemented to be used in high-throughput IoT applications such as NETCOM and BULKSTORE.
- This ASIC is the first hardware implementation offering the leakage-resilient authenticated encryption scheme ISAP [5], which is suitable for BULKSTORE. For NETCOM, it implements leakage-resilient encryption based on the 2PRG primitive [17], AES-128, and secure re-keying through a masked and shuffled polynomial multiplication.
- Through evaluation on a running ASIC, we demonstrate that integrating the cryptographic accelerator into the memory subsystem of a SoC allows us to reach side-channel protected encryption with a throughput of up to 7.49 Gbit/s in a conventional 65 nm CMOS process and within a power envelope of 100 mW.

The remainder of this paper is structured as follows. In Section II, we discuss the backgrounds of side-channel attacks and countermeasures. In Section III, we present the hardware architecture and design rationales of the proposed chip. Section IV evaluates the performance in terms of throughput and power consumption. Finally, in Section VI, we conclude of this work.

II. BACKGROUND

The execution of a cryptographic implementation leaks information on the processed data via various side-channels, such as the power consumption or the electromagnetic radiation. This information leakage is exploited by attackers in so-called side-channel attacks to learn secret information such as the encryption key. Independent of the concrete source of side-channel leakage, there are two basic types of side-channel attacks: simple power analysis (SPA) and differential power analysis (DPA). While SPA tries to recover the secret key of a cryptographic implementation from observing the power consumption (or any equivalent side-channel) during the en-/decryption of one single input, DPA uses observations during the en-/decryption of many different inputs. Hereby, DPA is particularly effective as there is more side-channel leakage available, the more data is processed under one single key.

A. Frequent Re-Keying

The probability for a key recovery via DPA to be successful rises with the number of side-channel observations for different inputs under the same key K. Therefore, one approach to counteract DPA is frequent re-keying [10], [12], where the goal is to design a cryptographic scheme such that for a certain key K, the number of different inputs to the underlying cryptographic primitive is upper-bounded by some small number q (q-limiting [17]). As soon as the limit of q different inputs is reached, another key K' is selected, which limits the data complexity per single key K. Thus, for a certain key K, the cryptographic implementation can only generate the side-channel leakage for q different inputs, which effectively limits the feasibility of DPA to recover K. As a result, the implementation of the cryptographic primitive is only required to resist SPA attacks.

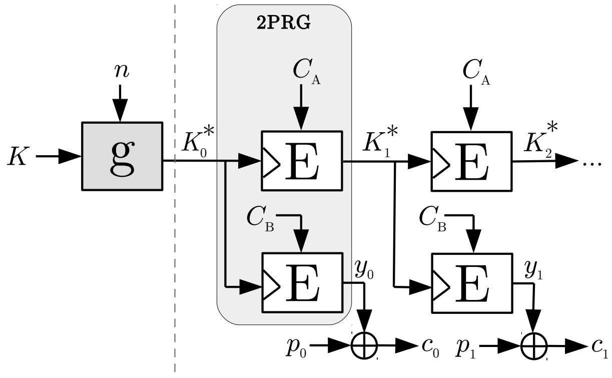

Fig. 1. 2PRG-based leakage-resilient stream cipher.

B. Leakage-Resilient Cryptography

One prominent example for this re-keying approach is leakage-resilient encryption [17], [16], such as depicted in Figure 1. This scheme consists of two basic parts: (1) a secure re-keying function g, and (2) a leakage-resilient encryption scheme. The re-keying function $g : (K, n) \mapsto K^*$ securely derives a fresh session key K^* from a pre-shared master secret K and a fresh nonce n and hence must be implemented such as to resist both SPA and DPA attacks. However, the choice of a fresh nonce n results in a fresh session key K^* to be used for each invocation of the leakage-resilient encryption scheme. The leakage-resilient encryption scheme, on the other hand, is designed such as to guarantee a data complexity $q = 2$ per key when performing the actual en-/decryption. For this purpose, the leakage-resilient encryption mode in Figure 1 utilizes a key update step $K_i^* \mapsto K_{i+1}^*$ to provide a different key for the encryption of each plaintext block p_i. More concretely, the 2PRG primitive used in Figure 1 encrypts two constant values C_A and C_B using a block cipher E with the input key K_i^* to give the next block's key K_{i+1}^* and a pad y_i. The pad y_i is used for en-/decrypting the respective p_i/c_i.

While modes like in Figure 1 provide confidentiality and side-channel security for devices that en-/decrypt data only a single time, such as in the communication setting NETCOM, Dobraunig et al. [5] point out that these modes remain vulnerable to DPA in scenarios supporting multiple decryptions of the same data, as it occurs for the bulk storage setting BULKSTORE. In particular, such scenarios

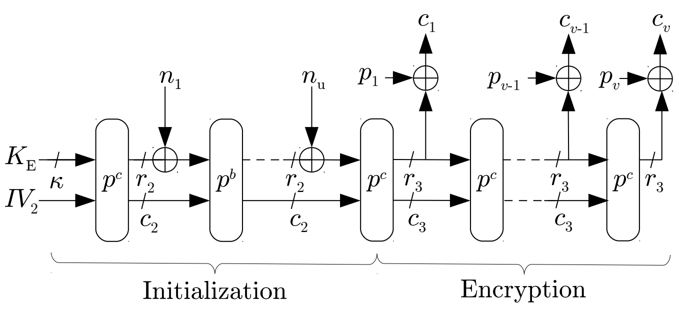

Fig. 2. IsapRk and IsapEnc.

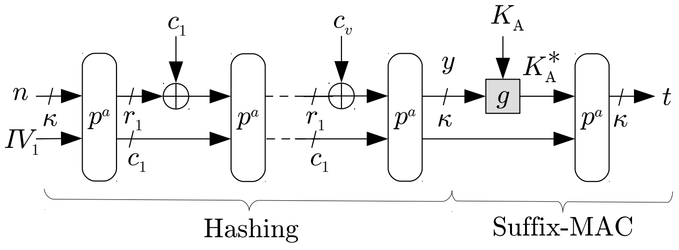

Fig. 3. IsapMac.

allow attackers to tamper with ciphertexts, to observe the respective decryption processes, and thus to perform a DPA. To provide side-channel security also in the setting BulkStore, Dobraunig et al. [5] further propose the authenticated encryption scheme Isap.

Isap is an encrypt-then-MAC scheme consisting of the two algorithms IsapEnc and IsapMac depicted in Figure 2 and 3 and using the keys K_E and K_A, respectively. IsapEnc uses a nonce n for re-keying during initialization and follows the same principle of continuous key updates during encryption as previous modes. However, in addition, IsapMac ensures DPA security for IsapEnc in case of malicious modifications of ciphertexts. In terms of IsapMac, the core idea to prevent DPA is to bind the MAC key K_A^* to the hash y of the ciphertexts. This automatically results in different keys for different ciphertexts. For the security level $\kappa = 128$ bits, IsapEnc and IsapMac use Keccak-f[400] as the permutation p with $a = 20$, $b = 12$, and $c = 12$ rounds, and the rates $r_1 = 144$, $r_2 = 1$, and $r_3 = 144$. With these parameters, injection of the nonce in IsapEnc is 2-limiting, and DPA is prevented. However, a side-channel secure re-keying function g is required for IsapMac to derive the session key K_A^* from the pre-shared authentication master key K_A.

C. Secure Re-Keying Function

There are several suitable designs for the re-keying function $g : (K, n) \mapsto K^*$ available. One prominent way is to build g such that it is easy to protect with classical countermeasures such as masking or shuffling. Medwed et al. propose a polynomial multiplication in a finite field of the key K and a nonce n. This multiplication can be masked and shuffled easily. However, as pointed out in [11], [2], [1], [9], [15], [4], the algebraic structure of a multiplication opens the door to combined attacks on g and the encryption. To mitigate this type of attack, Dobraunig et al. propose to add a block-cipher based feed-forward computation [6] after the multiplication.

Another approach for designing a secure re-keying function is exploiting a GGM construction [8] as proposed in [17], [7]. The GGM construction is a tree-like approach to mix a secret K with a public n, where on each tree level exactly one bit of the public n is evaluated. Starting with $s_0 = K$, the key s_{i+1} is computed by encrypting one of two predefined plaintexts p_0 or p_1 with the key s_i and block cipher E, depending on the i-th bit of n. The output of the last level is then, after a post-processing, used as the session key K^*. GGM-based re-keying is 2-limiting and hence considered to be secure against DPA. As a variant, [5] presents the sponge version of GGM-based re-keying, IsapRk. The construction of IsapRk is also embedded in IsapEnc and essentially comprises the nonce-absorbing initialization part in Figure 2, but truncates the output to give a κ-bit session key. This re-keying operation is reused as part of IsapMac to bind the hash of the message with the authentication key K_A.

III. Architecture

In order to provide leakage-resilient cryptography with high throughput on a limited power budget, we designed *Fulmine*, an ASIC implementing a large-scale IoT processor that embeds a hardware accelerator for leakage-resilient cryptography. This accelerator serves both use cases Netcom and BulkStore by integrating secure re-keying functions, the AES-based 2PRG primitive, and Isap. This system, as shown in Figure 4, implements a heterogeneous multi-core platform with two flexible software programmable accelerators. The SoC is organized in two distinct voltage and frequency domains, the soc, and the cluster domain. Both domains communicate via AXI4 interfaces with included dual-clock FIFOs and level shifters. The soc domain contains 192 kB level-2 memory for data and instructions, a 4 kB ROM, and different peripherals (UART, (quad) SPI, I2C, I2S, GPIO) including a peripheral DMA.

The cluster domain contains four general purpose programming units based on the OpenRISC instruction set enhanced with dedicated instructions for hardware accelerated loops and DSP operations. Furthermore, this domain includes two software programmable accelerators used for cryptography and image processing. All six processing units are connected via a logarithmic interconnect to the cluster internal 64 kB level-1 tightly-coupled data memory (TCDM). The logarithmic interconnect provides all units with parallel single-cycle access to the TCDM. If two units access the same TCDM memory block at the same time, a round-robin arbitration policy is used to determine which unit gets access and the other units are stalled. Sharing the memory between the general purpose processors and the custom accelerators optimizes the per-

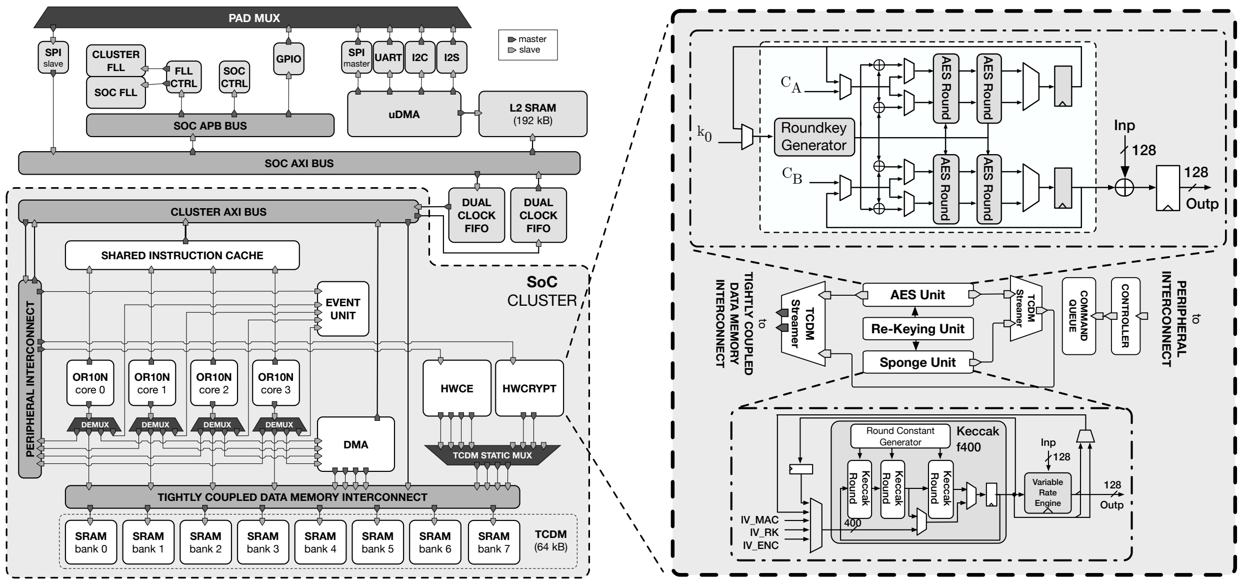

Fig. 4. SoC architecture with HWCRYPT details.

formance and increases the flexibility since the architecture avoids any point-to-point connections and memory copies.

A. Cryptographic Accelerator

HWCRYPT, as depicted in Figure 4, is a cryptographic accelerator, which operates in parallel with the general purpose computing units. This accelerator contains all the leakage-resilient modes required for the defined use cases. The accelerator contains two 32-bit TCDM interfaces for parallel read and write access to the memory, which are seamlessly integrated into the memory subsystem of the cluster, and a dedicated 32-bit memory-mapped configuration interface. The source and destination memory address, length, operating mode of *HWCRYPT* can be set by any processor in the cluster via memory-mapped registers. The configuration interface allows the processors to start and monitor the operation via polling, or alternatively, an event/interrupt unit can wake up the processor once the operation has finished. To increase the utilization and therefore the overall performance, *HWCRYPT* supports a command queue with five pending configurations to support an asymmetric configuration while the accelerator is still busy. When one encryption operation finishes, the accelerator looks into the command queue and automatically starts a new job if there is a pending configuration entry. The interrupt configuration supports intermediate IRQs when a single job finishes and a final IRQ when the command queue gets empty.

The heart of the *HWCRYPT* cryptographic accelerator is the *AES Unit* and *Sponge Unit* responsible for encryption, and a flexible re-keying unit. These building blocks are surrounded by an input and output TCDM streaming unit required for the block size conversion between the 32-bit memory interfaces and the 128-bit blocks used by the encryption units. Furthermore, memory mapped control registers allow the processing cores to configure the accelerator for the encryption easily.

The *AES Unit* implements the 2PRG-based stream cipher in Figure 1 using AES-128. Computing the encryption pad for one message block and updating the key requires two invocations of AES-128. Since these computations do not have a data dependency, they can be parallelized. For this reason, the hardware design contains two instances of AES-128, which share the same AES key scheduling unit. Both AES instances are implemented fully in parallel with two AES rounds in the combinational path. Since this architecture can execute two AES-rounds within one clock cycle, executing all ten rounds takes five clock cycles plus one cycle of preloading the input register. Apart from the leakage-resilient mode in Figure 1, the *AES Unit* provides a special operation mode to give the CPU cores direct access to the AES round function, which can be used to accelerate any cryptographic algorithm that relies on the AES round function.

The second major block within *HWCRYPT* is the *Sponge Unit*, which is internally based on the KECCAK-$f[400]$ permutation. This permutation is implemented fully in parallel with three rounds in the combinational path. The number of processed rounds is configurable in multiples of three with a maximum of 20 processed round operations. This engine supports all operating modes of ISAP as specified in Section II, including the side-channel resistant re-keying function ISAPRK, the encryption scheme ISAPENC, and ISAPMAC for ciphertext authenticity. However, for each stage of the operation, the *Sponge Unit* can be configured with different block sizes (1 bit up to 128 bits in powers of two) and a different number of rounds, which allows the user to trade off between performance and security. Instead

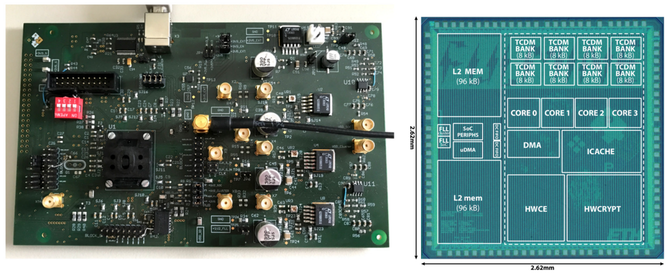

Fig. 5. Measurement setup and Fulmine chip micrograph with highlighted operational blocks.

of using a rate of 144 bits for feeding in the plain/ciphertext, we chose a rate of 128 bit to be compatible with the block size of the 2PRG. Reducing the rate slightly reduces the throughput but also increases the security. Similar to the *AES Unit*, the *Sponge Unit* also supports low-level access to the underlying primitive to accelerate software implementations of algorithms using the KECCAK-f[400] permutation.

The re-keying unit in Figure 4 takes a master key K and a fresh nonce n from the configuration registers and computes a session key $K^* = K \cdot n$, where $\cdot$ denotes a polynomial multiplication in $GF(2^8)[y](y^{16}+1)$ as proposed by Medwed et al [12]. The polynomial multiplication algorithm is transformed to the operand scan form, which is better suited for a parallel implementation. To achieve a parallel implementation, the re-keying unit contains 16 instances of a $GF(2^8)$ multiplier. Given this partial parallel architecture, computing one session key takes 18 clock cycles, including the time for configuring all registers. The re-keying unit offers side-channel protection by means of additive masking (with a configurable masking order) and shuffling of the partial products. In addition to shuffling the start index as proposed in [12], this architecture supports the shuffling of all partial products resulting in 16! different shuffling sequences. Note that shuffling does not decrease the multiplication throughput of our implementation. However, masking decreases the performance since our architecture only contains one polynomial multiplier. Concretely, computing a masked session key takes $18 \cdot (d+1)$ clock cycles, where d denotes the masking order. Both, the masking and shuffling unit get their required randomness from a shared pseudo-random number generator, which security is not scope of this work.

Furthermore, the re-keying unit performing polynomial multiplication also supports a post-processing step comprising a feed-forward branch of the master key and a block cipher call as proposed by Dobraunig et al. [6], which mitigates key recovery through time-memory trade-off attacks such as in [4]. To achieve this task, we reuse AES-128 from the *AES Unit* and add a static overhead of 20 clock cycles to the re-keying operation.

IV. Evaluation

In this section, we evaluate the performance and efficiency of our ASIC in the use cases NETCOM and BULKSTORE. The prototype chips were fabricated in the UMC 65 nm LL 1P8M technology. Figure 5 shows the measurement setup and the annotated chip micrograph of the fabricated Fulmine chip. *HWCRYPT*'s area equals 349 kGE from which 63 % is used by the *AES Unit*, 17 % by the *Sponge Unit*, and 5.5 % by the polynomial re-keying unit. The rest of the area is used for bus interfaces, control logic, and glue logic. The evaluation is performed between the supported voltage range of the CLUSTER domain between 0.8 V and 1.2 V. We evaluate the raw encryption or streaming performance, the throughput of the re-keying functions, and the leakage-resilient protocols combining the first two operations.

NETCOM requires high-throughput leakage-resilient encryption but does not involve multiple decryptions. Therefore, encryption based on the 2PRG using AES-128 is suitable for this use case. For this setup, our evaluation, as summarized in Table I, shows a maximum throughput of 5.39 Gbit/s at a frequency of 256 MHz, which equals an architectural throughput of 0.38 cycles per byte (cpb). On the other hand, BULKSTORE requires support for securely performing multiple decryptions. Hence, ISAP is suitable for this use case. The raw encryption performance of ISAPENC reaches 7.49 Gbit/s at a maximum frequency of 356 MHz. Combining the raw encryption in ISAPENC with the hashing part of ISAPMAC results in a raw streaming performance of 2.95 Gbit/s.

Both encryption units reach a raw encryption performance (without the re-keying or MAC steps) of 0.38 cycles per byte (cpb), which is almost twice as fast as the fastest AES-ECB encryption using the Intel AES-NI instruction set of modern desktop processors [3]. The best-case energy efficiency of our design reaches 137 Gbit/s/W, making it suitable for various IoT applications.

Both leakage-resilient encryption modes use a re-keying function g for their initialization. The 2PRG-based mode uses a side-channel protected polynomial multiplication, and ISAPENC uses ISAPRK, the sponge variant of a GGM-based re-keying function. The two re-keying functions are implemented in hardware, and we evaluate them both in isolation and in combination with their corresponding encryption mode. Table II summarizes the performance of the sole re-keying functions regarding the required computing cycles. The re-keying function based on the polynomial multiplication employs first-order masking together with shuffling of

TABLE I
ENCRYPTION PERFORMANCE.

Mode	Operating Point [V, MHz]	Power[a] [mW]	Performance [Gbit/s]	Efficiency [Gbit/s/W]
2PRG	0.8, 85	24	1.79	74.6
	1.2, 256	153	5.39	56.1
ISAP	0.8, 85	13	1.79	137.7
	1.2, 356	96	7.49	78

[a]Only includes the power consumption of the cluster.

all indices of the partial multiplications. IsapRk uses more cycles since it requires one permutation invocation for each bit of the nonce (144 bits used in IsapRk). However, since the re-keying functions are only used during initialization or as part of the suffix-MAC, their performance impact diminishes asymptotically with the message length.

Combining the re-keying function with the streaming performance of the encryption mode yields the overall throughput for our use cases. To measure this performance, we selected a block size of 8 kB, which is reasonable given that Ethernet jumbo frames can take up to 8960 bytes of TCP payload and storage devices use a block size in the same order of magnitude to partition the memory. For Netcom using this block size, the encryption throughput using the 2PRG-based mode is 5.29 Gbit/s including the static overhead of first-order masked re-keying. For BulkStore, Isap as a combination of IsapRk, IsapEnc, and IsapMac reaches a maximum throughput of 2.69 Gbit/s. This mode however also provides ciphertext authenticity on top of confidentiality.

Summarizing, our evaluation shows that the implemented cryptographic accelerator yields high throughput and energy efficiency. In addition, the accelerator offers flexible options for configuration, making it suitable for a variety of different high-performance applications including Netcom and BulkStore. Further measurements will analyze the side-channels security of the DPA-secure re-keying functions.

V. Acknowledgement

This project has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (grant agreement No 681402).

VI. Conclusion

In this work, we present a fabricated ASIC implementation of a multi-core SoC targeted for the IoT enhanced with a high-speed leakage-resilient cryptographic accelerator supporting different modes of operation. The accelerator integrates the first hardware implementation of Isap and a leakage-resilient 2PRG stream cipher together with a side-channel protected re-keying function based on a polynomial multiplication. Performance evaluation shows that the accelerator can reach a throughput of up to 7.49 Gbit/s making it suitable for a variety of high-performance applications. Furthermore, we reach an energy efficiency of up to 137 Gbit/s/W. Integrating the cryptographic accelerator into the memory subsystem of the SoC allows us to build high-throughput leakage-resilient primitives meeting high-performance requirements of today's IoT applications.

TABLE II
Re-keying Performance.

Re-keying Mode	Nonce / bits	Cycles
Polynomial Re-keying[a]	128	56
IsapRk	144	739

[a]First order masking with full shuffling and post-processing. Increasing the masking order adds 18 cycles per order to the total duration

References

[1] S. Belaïd, J. Coron, P. Fouque, B. Gérard, J. Kammerer, and E. Prouff, "Improved side-channel analysis of finite-field multiplication," in *Cryptographic Hardware and Embedded Systems - CHES 2015*, pp. 395–415.

[2] S. Belaïd, P. Fouque, and B. Gérard, "Side-channel analysis of multiplications in GF(2128) - application to AES-GCM," in *Advances in Cryptology - ASIACRYPT 2014*, 2014.

[3] A. Bogdanov, M. M. Lauridsen, and E. Tischhauser, "Comb to pipeline: Fast software encryption revisited," in *Fast Software Encryption - 22nd International Workshop, FSE 2015*, 2015, pp. 150–171.

[4] C. Dobraunig, M. Eichlseder, S. Mangard, and F. Mendel, "On the security of fresh re-keying to counteract side-channel and fault attacks," in *Smart Card Research and Advanced Applications - 13th International Conference, CARDIS 2014*, 2014.

[5] C. Dobraunig, M. Eichlseder, S. Mangard, F. Mendel, and T. Unterluggauer, "ISAP - towards side-channel secure authenticated encryption," *IACR Trans. Symmetric Cryptol.*, 2017.

[6] C. Dobraunig, F. Koeune, S. Mangard, F. Mendel, and F. Standaert, "Towards fresh and hybrid re-keying schemes with beyond birthday security," in *Smart Card Research and Advanced Applications - 14th International Conference, CARDIS 2015*, 2015.

[7] S. Faust, K. Pietrzak, and J. Schipper, "Practical leakage-resilient symmetric cryptography," in *Cryptographic Hardware and Embedded Systems - CHES 2012*, 2012.

[8] O. Goldreich, S. Goldwasser, and S. Micali, "How to construct random functions," *J. ACM*, 1986.

[9] Q. Guo and T. Johansson, "A new birthday-type algorithm for attacking the fresh re-keying countermeasure," *IACR Cryptology ePrint Archive*, 2016.

[10] P. Kocher, "Leak-resistant cryptographic indexed key update," Mar. 25 2003, uS Patent 6,539,092. [Online]. Available: https://www.google.com/patents/US6539092

[11] M. Medwed, C. Petit, F. Regazzoni, M. Renauld, and F. Standaert, "Fresh re-keying II: securing multiple parties against side-channel and fault attacks," in *Smart Card Research and Advanced Applications - CARDIS 2011*.

[12] M. Medwed, F. Standaert, J. Großschädl, and F. Regazzoni, "Fresh re-keying: Security against side-channel and fault attacks for low-cost devices," in *Progress in Cryptology - AFRICACRYPT 2010, Third International Conference on Cryptology in Africa. Proceedings*, 2010.

[13] T. Nakai, M. Shibatani, M. Shiozaki, T. Kubota, and T. Fujino, "Side-channel attack resistant AES cryptographic circuits with ROM reducing address-dependent EM leaks," in *IEEE International Symposium on Circuits and Systems, ISCAS 2014*, 2014.

[14] Y. Peng, H. Zhao, X. Sun, and C. Sun, "A side-channel attack resistant AES with 500mbps, 1.92pj/bit PVT variation tolerant true random number generator," in *2017 IEEE Computer Society Annual Symposium on VLSI, ISVLSI 2017*, 2017.

[15] P. Pessl and S. Mangard, "Enhancing side-channel analysis of binary-field multiplication with bit reliability," in *Topics in Cryptology - CT-RSA 2016 - The Cryptographers' Track at the RSA Conference 2016*, 2016.

[16] K. Pietrzak, "A leakage-resilient mode of operation," in *Advances in Cryptology - EUROCRYPT 2009, 28th Annual International Conference on the Theory and Applications of Cryptographic Techniques*, 2009.

[17] F. Standaert, O. Pereira, Y. Yu, J. Quisquater, M. Yung, and E. Oswald, "Leakage resilient cryptography in practice," in *Towards Hardware-Intrinsic Security - Foundations and Practice*, 2010.