# Comparison of Cost of Protection Against Differential Power Analysis of Selected Authenticated Ciphers[1]

William Diehl, Abubakr Abdulgadir, Farnoud Farahmand, Jens-Peter Kaps and Kris Gaj

George Mason University, Fairfax VA 22033, USA
{wdiehl,aabdulga, ffarahma, jkaps, kgaj}@gmu.edu

**Abstract.** Authenticated ciphers, like all physical implementations of cryptography, are vulnerable to side-channel attacks, including differential power analysis (DPA). The t-test leakage detection methodology has been used to verify improved resistance of block ciphers to DPA after application of countermeasures. However, extension of the t-test methodology to authenticated ciphers is non-trivial, since authenticated ciphers require additional input and output conditions, complex interfaces, and long test vectors interlaced with protocol necessary to describe authenticated cipher operations. In this research we augment an existing side-channel analysis architecture (FOBOS) with t-test leakage detection for authenticated ciphers. We use this capability to show that implementations in the Spartan-6 FPGA of the CAESAR Round 3 candidates ACORN, ASCON, CLOC (AES and TWINE), SILC (AES, PRESENT, and LED), JAMBU (AES and SIMON), and Ketje Jr., as well as AES-GCM, are vulnerable to 1st order DPA. We then implement versions of the above ciphers, protected against 1st order DPA, using threshold implementations. The t-test leakage detection methodology is used to verify improved resistance to 1st order DPA of the protected cipher implementations. Finally, we benchmark unprotected and protected cipher implementations in the Spartan-6 FPGA, and compare the costs of 1st order DPA protection in terms of area, frequency, throughput, throughput-to-area (TP/A) ratio, power, and energy-per-bit. Our results show that ACORN has the lowest area (in LUTs), the highest TP/A ratio, and is the most energy-efficient of all DPA-resistant implementations. However, Ketje Jr. has the highest throughput.

**Keywords:** Cryptography, authenticated cipher, field programmable gate array, power analysis, side channel attack, countermeasure, lightweight, t-test

## 1 Introduction

Today's environment of large and high-speed centralized cloud-based computing is expanding into tomorrow's smaller and lightweight, "edge-based computing," which will consist of billions of devices in the "Internet of Things" (IoT). IoT devices, while heavily constrained by size, weight, and power (SWaP) considerations, are particularly vulnerable to cyber-security threats, since they often reside physically apart from secure data facilities. Authenticated ciphers, such as AES-GCM, are well-suited for lightweight edge devices in the IoT, since they combine the functionality of confidentiality, integrity, and authentication services, and can potentially provide the same security as a conventional cipher combined with message authentication code at reduced cost.

Cryptographic algorithms, which have been subjected to research, analysis, and public

[1] This is an extended version of a paper of the same title accepted for publication at HOST 2018 (© 2018 IEEE).

scrutiny, are generally secure against cryptanalysis given the capabilities of current computing, in that the best-known cryptanalytic attacks are no easier than a "brute-force" attack. However, actual ciphers exist in the physical world and are implemented in imperfect devices, which can be exploited by analyzing physical phenomena through "side channel attacks" such as differential power analysis (DPA) to recover all or part of sensitive variables.

The Competition for Authenticated Encryption: Security, Applicability, and Robustness (CAESAR), seeks to identify a portfolio of authenticated ciphers that offer advantages over AES-GCM, and are suitable for widespread adoption [1]. The CAESAR committee specified use-cases for which candidates would be optimized and ultimately selected during Round 3 and the Final Round. One of these use cases is for lightweight applications (resource constrained environments), for which desired characteristics include "natural ability to protect against side-channel attacks" [2]. Accordingly, it is desirable to examine implementations of CAESAR candidates intended for lightweight applications to determine resistance of unprotected and protected implementations to DPA, and the costs of protection when required. However, to date, there has been no study of the side-channel resistance of multiple authenticated ciphers, implemented using the same methodology and same test equipment, and no study of the comparative costs of protection against DPA.

In this work, we demonstrate a methodology for analyzing authenticated ciphers for vulnerabilities to power analysis side-channel attack, and evaluation of the effectiveness of countermeasures. We leverage an existing t-test leakage detection methodology [3], and upgrade the Flexible Open-source workBench fOr Side-channel analysis (FOBOS) [4], to perform t-tests on authenticated ciphers. The FOBOS interface with the victim cipher implementation is standardized by leveraging the CAESAR Hardware Applications Programming Interface (API) for Authenticated Ciphers, which was adopted by the CAESAR committee in May 2016 [5, 6]. Additionally, the use of the Development Package for the CAESAR Hardware API, available at [7], facilitates a repeatable and exportable test methodology for all CAESAR candidates.

Using the augmented FOBOS, we demonstrate t-tests on eleven unprotected authenticated ciphers, implemented on the test device (Spartan 6 FPGA), including AES-GCM, ACORN, ASCON, CLOC (AES, TWINE), SILC (AES, PRESENT, LED), JAMBU (AES, SIMON), and Ketje Jr. After demonstrating vulnerabilities to DPA, we upgrade the same ciphers using threshold implementation (TI)-protection, and verify improved resistance to DPA.

Additionally, through analysis of CLOC-AES and CLOC-TWINE, we identify a limitation of the t-test's ability to find DPA vulnerability. We demonstrate this limitation by analysis of a data-dependent conditional decision in the CLOC specification, which leads to a failed t-test, but does not expose secret key, or any data not known to the observer a priori. Finally, we use the augmented FOBOS architecture to perform power analysis of the ciphers during operation on the Spartan 6 FPGA, using representative test vectors. The resulting unprotected and protected ciphers are compared in terms of FPGA resources (LUTs and slices), maximum frequency (MHz), throughput (Mbps), throughput-to-area (TP/A) ratio (Mbps/LUT), power (mW), and energy-per-bit (nJ/bit), in order to determine absolute and relative costs of protection.

## 2 Background and Previous Work

### 2.1 Authenticated Ciphers

Authenticated Ciphers incorporate the functionality of confidentiality, integrity, and authentication. Input to authenticated ciphers consists of such fields as $Message$, associated data $AD$ (which may include, for example, a header or trailer of a packet used in

communication protocols), a secret $Key$, and a public message number $Npub$. In authenticated encryption, a $Ciphertext$ is computed as a function of the inputs, ensuring the confidentiality of the transaction. A $Tag$, which is a function of all blocks of $AD$, $Message$, $Npub$, and $Key$, is produced at the conclusion of message encryption, and assures integrity and authenticity of the transaction. In authenticated decryption, $Ciphertext$ is decrypted to $Message$, and $Tag'$ is typically computed as a function of $Ciphertext$, $AD$, $Npub$, and $Key$. If $Tag = Tag'$ then authentication and integrity of the transaction are assured; otherwise the decrypted $Ciphertext$ is not released. If authenticity and integrity are verified, the outputs of the transaction are $AD$ and $Message$ [8]. A notional authenticated cipher is shown in Fig. 1.

In this research, we analyze the CAESAR Round 3 variants of the ACORN, ASCON, CLOC-SILC, JAMBU, and Ketje families of authenticated ciphers, described in [9 – 13], respectively. We choose these ciphers since their authors have specified an intended lightweight use case for their respective ciphers. We additionally analyze the existing defacto standard AES-GCM, described in [14], for purposes of comparison. We use RTL VHDL implementations of AES-GCM, ASCON, CLOC-AES, JAMBU-AES, and SILC-AES available at [15], ACORN at [16], CLOC-TWINE, SILC-PRESENT, and SILC-LED at [17], JAMBU-SIMON available at [18], and Ketje Jr. available at [19]. However, we modify both the unprotected and protected implementations of the above ciphers as necessary to achieve implementations protected against 1st order DPA, and facilitate fair benchmarking comparisons. The authenticated ciphers investigated in this research, including relevant characteristics, are summarized in Table 1.

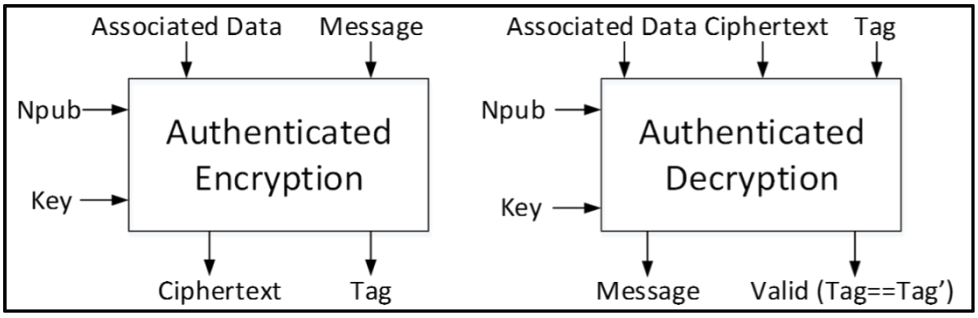


Fig. 1. Notional Authenticated Cipher

Table 1. Authenticated Ciphers Examined in this Research

| Authenticated Cipher | Spec | Implementation | Key Size (bits) | Block Size (bits) | Tag Size (bits) |
|---|---|---|---|---|---|
| AES-GCM | [14] | CERG GMU [15] | 128 | 128 | 128 |
| ACORN | [9] | CCRG NTU [16] | 128 | 8 | 128 |
| ASCON | [10] | CERG GMU [15] | 128 | 64 | 128 |
| CLOC-AES | [11] | CERG GMU [15] | 128 | 128 | 64 |
| CLOC-TWINE | [11] | CLOC-SILC Team [17] | 80 | 64 | 32 |
| SILC-AES | [11] | CERG GMU [15] | 128 | 128 | 64 |
| SILC-PRESENT | [11] | CLOC-SILC Team [17] | 80 | 64 | 32 |
| SILC-LED | [11] | CLOC-SILC Team [17] | 80 | 64 | 32 |
| JAMBU-AES | [12] | CERG GMU [15] | 128 | 64 | 64 |
| JAMBU-SIMON | [12] | CCRG NTU [18] | 96 | 48 | 48 |
| Ketje Jr. | [13] | Ketje-Keyak Team [19] | 96 | 32 | 64 |

### 2.2 Leakage Detection Methodology: Welch's T-test

Differential Power Analysis (DPA) is used to analyze differences between observed power measurements, and hypothetical power (based on presumed contents of a sensitive variable) according to a power model. However, coming up with a power model is difficult, time

consuming, and requires expert knowledge of the underlying architecture [20 – 22].

An expedited leakage assessment methodology proposed in [3] and further described in [23], uses the Welch's t-test to determine whether two distributions are different from one another. Some of the advantages in using the t-test for an assessment of leakage are that it finds leakage of information without mounting an attack, does not rely on knowledge of the underlying architecture, and can quickly reveal when the information leaks and when a countermeasure has failed. However, it is not a complete substitution for DPA. For example, there is no recovery of a key, message, sensitive intermediate values, or the correct power model, and no information is gained about the difficulty of mounting an attack.

The t-test is the Welch's t-test, in which a confidence factor $t$ is calculated as

$$t = (\mu_0 - \mu_1)/\sqrt{{s_0}^2/n_0 + {s_1}^2/n_1} \quad (1)$$

where $\mu_0$ and $\mu_1$ are means of distributions $Q_0$ and $Q_1$ (to be subsequently defined), $s_0$ and $s_1$ are standard deviations, and $n_0$ and $n_1$ are the cardinality of the distributions, or the number of samples.

Given a normally distributed probability density function (pdf) $f(t)$, a probability of accepting a null hypothesis $p$ is calculated as $p = 2\int_{|t|}^{\infty} f(t)dt$. To use the t-test, we start with two distributions, and assume a null hypothesis – namely, that "samples are drawn from the same distribution." and that "samples are not distinguishable." We designate a "threshold," e.g.,$|t| > 4.5$, beyond which we reject the null hypothesis. If this occurs during analysis of the two distributions, we reject the null hypothesis that "the samples are from the same distribution" and reason that the device is leaking information.

If our goal is to plausibly show that a device is leaking information (without a specific need to recover a sensitive variable or demonstrate the difficulty of an attack), we can use the so-called "non-specific t-test." In the non-specific t-test, we preselect some "fixed" input data $D$ (e.g., $Message$, $AD$, $Npub$). Then we randomly interleave the feeding of $D$, or random data, to the algorithm. We call this characterization a "fixed versus random" test [22, 23]. The above method has been used to show vulnerabilities in block ciphers, and to confirm the effectiveness of countermeasures to DPA (e.g., [22, 24, 25]). Additionally, the t-test has been used to evaluate countermeasures in ASCON, although the authors do not discuss how such a methodology would be exportable to other CAESAR candidate ciphers [26].

### 2.3 Threshold Implementations

Threshold implementations, or TI, are an algorithmic countermeasure against power-analysis side-channel attack. TI are based on secret sharing and multi-party communications, where the communications of a single party cannot be exploited to learn the secret content [27 – 29].

TI improve upon traditional Boolean masking in that they provide security in the presence of glitches. Although Boolean masking provides mathematically-secure protection against DPA, it can fail in CMOS technology, since the power change that occurs in a CMOS gate during a transition due to a glitch is relatively large compared to normal operation of a device. Measuring the toggle rate of CMOS glitches has been used to successfully attack a masked version of AES [30].

A threshold implementation must have the following three properties, outlined in [27], to be provably secure against power analysis in the presence of glitches:

1. Non-completeness. Every function is independent of at least one share of each of the input variables. Defined formally, if $z = F(x, y)$, and $x$ and $y$ are divided into $n$ shares, then

$$z_1 = f_1(x_2, x_3, \dots, x_n, y_2, y_3, \dots, y_n), \quad (2a)$$
$$z_2 = f_2(x_1, x_3, \dots, x_n, y_1, y_3, \dots, y_n), \quad (2b)$$

$$\vdots$$

$$z_n = f_n(x_1, x_2, \dots, x_{n-1}, y_1, y_2, \dots, y_{n-1}). \quad (2c)$$

In other words, If $z_i$ does not depend on $x_i$ and $y_i$, it cannot leak information about $x_i$ or $y_i$.

2. Correctness. The sum of the output shares gives the desired output. Formally

$$z = \oplus_{i=1}^{n} z_i = N(x, y). \quad (3)$$

3. Uniformity. A realization of sharing $z = F(x, y)$ is uniform if for all distributions of the inputs $a$ and $b$, the output distribution preserves the input distribution. In other words, if the input function is a permutation, the output function should also be a permutation.

A non-linear function of algebraic degree 2, such as z= $xy$ (e.g., a 2-input `and` gate), can be shared using three TI shares, since $d + 1$ shares are required to share a function of degree $d$. However, as discussed in [31, 32], achieving the TI uniformity property is not trivial. This property can be achieved by supplying fresh random bits (e.g., "resharing" or "remasking" randomness), however, this requires the resourcing of sufficient randomness, which must either be imported into the device, or generated internally at run-time. Thus, the decision to use 3-share TI which require an increased number of random bits, or 4-share TI with more required resources but no additional randomness, is an engineering design tradeoff.

### 2.4 Our contribution

In this research, we expand upon the methodology in [22] to provide the first documented methodology suitable for analyzing side-channel resistance, and evaluating the effectiveness of countermeasures against side-channel attacks (SCA), for a large number of authenticated ciphers (e.g., there are 15 CAESAR Round 3 candidates, not including multiple variants of some ciphers). Our methodology uses a free and open-source SCA test bench (FOBOS), published specification for the CAESAR Hardware API for Authenticated Ciphers, associated Development Package, and publicly-available source codes for the unprotected cipher implementations in this research. As such, it should be possible for other researchers to either duplicate, or improve upon these results.

We also demonstrate one of the limits of the t-test leakage detection methodology to identify DPA vulnerability by showing a case of a "false positive" t-test based on a data-dependent conditional decision in the CLOC cipher.

Finally, it is well-known that the implementation of countermeasures against DPA is costly, in terms of resources and performance. However, comparison between multiple ciphers often occurs using ambiguous metrics, performed by diverse research groups, and operating on different hardware and test architectures. We illustrate a methodology for the comparison of the costs of protection against 1st order DPA which are suitable for adaptation across all authenticated ciphers, and could assist the CAESAR committee in selection of final round and final portfolio candidates.

## 3 Methodology

### 3.1 Leakage Detection Methodology for Authenticated Ciphers

1. Problem Statement

In order to conduct a fixed versus random t-test, we instantiate the cipher on a physical device (e.g., FPGA or microcontroller), isolate external noise sources, monitor changes in voltage or current that occur in response to varying input, capture data from thousands of repetitive traces, and perform off-line statistical analysis to diagnose vulnerabilities.

In order to avoid noise and corrupted analysis, we wish to prevent external I/O during trace collection. Additionally, we require test vectors, such as message and secret key, which reflect a fixed-versus-random methodology, are suitable for thousands of repetitions, and are available at the cipher module at the start of every trigger event. These conditions are easily met for the typical block cipher, where there are only a few (e.g., 16) bytes each of message and key for every trace event. These few bytes of data are stored in the cipher module itself prior to trigger, or in a thin-veneer of buffers on the test board. Additionally, the cipher-test architecture interface is typically trivial, consisting of (for example), *m*-bit message, *n*-bit secret key, *p*-bit ciphertext ports, clock, and control signals. Likewise, the only protocol events for block cipher operations are typically "start" and "done." As a result, it is usually easy for cipher developers to send their designs to a "power analysis test shop," and assume that the tester will be able to adapt the block cipher to their test architecture.

The above assumptions, however, do not hold for authenticated ciphers. In order to detect all possible leakage in an authenticated cipher, one should test a variety of sequences of operations, including key initialization, $Npub$ and $AD$ processing, authenticated encryption and decryption, and $Tag$ generation and verification. This requires a test vector potentially thousands of bytes long, interlaced with protocol that describe the entire range of permitted authenticated cipher operations. A sample authenticated cipher test vector is shown in Fig. 2.

```
70000000 20000000 D6000008 B0B1B2B3B4B5B6B7 12000000 43000000
Activate Key  Auth Enc  Npub Header          Npub          Null AD  Null PT
30000000 D6000008 B0B1B2B3B4B5B6B7 12000000 52000000 83000008
Auth Dec Npub Header          Npub          Null AD  Null CT  Tag Header
CA93BC236B91C12E 70000000 20000000 D2000008 B0B1B2B3B4B5B6B7
      Tag          Activate Key  Auth Enc  Npub Header       Npub
```

Fig. 2. Sample authenticated cipher test vector

This long test vector must be provided to the victim board (but remain outside the cipher) prior to the trigger event. In contrast to the block cipher, long test vectors will arrive and depart the cipher unit during the trace event, but should not enter or exit the victim board during the event. Additionally, an authenticated cipher must have a more-complex external interface to encompass the range of possible operations. It is not reasonable to expect that a laboratory engineer could adapt each individual custom-designed authenticated cipher interface to a power analysis test bench; and if so, the expense in time and resources would preclude performing a large-scale analysis of DPA-resistance of multiple authenticated ciphers.

2. Solution

Our solution is facilitated by the CAESAR committee's adoption of the CAESAR HW API for Authenticated Ciphers. Available at [5, 6], this API defines a protocol for all necessary authenticated cipher operations, as summarized above. The API also specifies an AXI-compatible external interface, shown in Fig. 3, and further described in [33]. Additionally, the Development Package for the CAESAR HW API contains a test vector generator, aeadtvgen.py, which generates predictable and comprehensive test vectors adequate for power analysis testing [7].

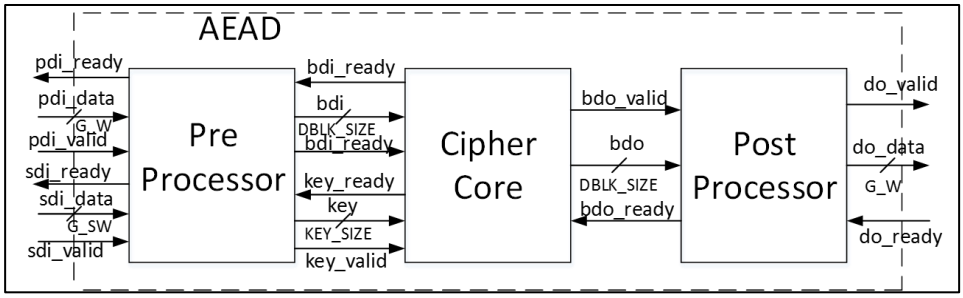


Fig. 3. External interface of the authenticated cipher module (AEAD), compliant with the CAESAR Hardware API [5, 6], and the internal top-level block diagram of AEAD supported by the Development Package [7].

We adapt the Flexible Open-source workBench fOr Side-channel analysis (FOBOS) to perform t-tests for side-channel leakage detection on authenticated ciphers. FOBOS is a single "acquisition to analysis" solution to measure resistance to power analysis side-channel attack (SCA) and evaluate the effectiveness of countermeasures [4]. It is trigger-activated, captures power analysis data in a specified window using an oscilloscope, and stores data offline for post-run analysis in a personal computer (PC). FOBOS uses a separate control board and victim board, where the Device Under Test (DUT), or "victim," is instantiated in the victim board.

In our instance of FOBOS, the oscilloscope used is the Agilent Technologies DSO6054A, and the control and victim boards are the Digilent Nexys-3 with Xilinx Spartan 6 FPGA. However, the components of FOBOS are built in a modular fashion so that the entire experimental setup can easily be adapted for different control and victim FPGA boards, oscilloscopes, and attack techniques.

For authenticated ciphers, the FOBOS DUT victim wrapper is configured with separate FIFOs corresponding to the data ports prescribed in [5], including public data interface (`pdi`), secret data interface (`sdi`), and data output (`do`). A fourth FIFO is aligned to the random data interface (`rdi`), which augments [5] to provide random data necessary for initial masking of public and secret data in protected ciphers. The FOBOS architecture, updated for authenticated ciphers, is shown in Fig. 4. The baseline FOBOS software suite, including acquisition and off-line side-channel analysis packages, is coded in Python and is available for download at [34].

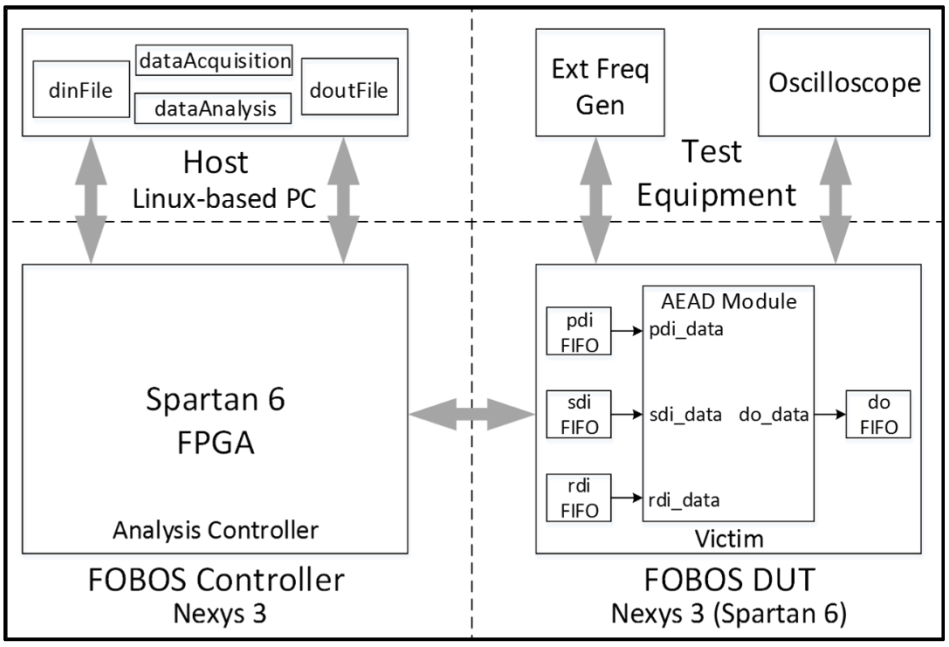


Fig. 4. FOBOS architecture modified for t-test leakage detection on authenticated ciphers

The procedure for performing t-tests on authenticated ciphers using FOBOS is summarized below:

(1) The test vector "dinFile.txt", created by aeadtvgen.py, is pre-formatted using a FOBOS parsing utility. It contains thousands of consecutive vectors of randomly-interleaved fixed or "random" data, where random data is substituted for all instances of $Npub$, $AD$, $Message$, $Ciphertext$, and $Tag$. The test vectors are wrapped in a layer of FOBOS-specific protocol, which determines their FIFO address on the victim board.

(2) Two separate bitstreams, FOBOS Controller (control board), and FOBOS DUT (which contains FOBOS DUT wrapper and victim cipher) are instantiated in hardware.

(3) The acquisition process (dataAcquisiton.py) is run from the PC. Each vector is loaded by the FOBOS Controller into FOBOS DUT. FOBOS Controller provides an oscilloscope trigger upon completion of test vector loading. Power measurements, sensed by a current probe and measured in the oscilloscope, are sent to the PC for off-line analysis. Data output (e.g., ciphertext) from each trace is accumulated in "doutFile.txt." Output data, although not used in the non-specific t-test, is valuable for ensuring proper cipher operation.

(4) At the completion of all traces, the tester performs off-line analysis on traces, stored in .npy format [35]. A utility routine "splits" the collected power traces into two distributions $Q_0$ and $Q_1$, according to a "fixed-versus-random" metafile created during test vector generation. The tester then runs the t-test utility on distributions $Q_0$ and $Q_1$, which generates a two-dimensional display of samples (corresponding to the time domain on the x-axis), and t-values, where sustained and repeatable results of $|t| > 4.5$ are considered a sign of vulnerability to DPA leakage.

### 3.2 Power and Energy Measurement

Measurement or computation of power and energy usage of a given authenticated cipher is desirable for multiple reasons. For example, mean power provides a figure of merit for required cooling capacity, while maximum power influences the choice of power supply and required conductor sizes. Total energy determines either the battery lifetime or cost of

electrical energy, and energy-per-bit (e.g. nJ/bit, where "bit" can be defined as one bit out of an $n$-bit block in an authenticated encryption or decryption) is a relative measure of energy efficiency for comparing different ciphers.

FPGA designers (e.g., Xilinx or Intel) provide an array of tools to estimate power consumption with incrementally increasing accuracy, depending on fidelity of model and level of effort. For example, Xilinx Power Analyzer (Xilinx ISE) or Xilinx Vivado Power Analyzer (Xilinx Vivado) can provide power estimates after a completed implementation, with no further input from the designer, using "vectorless" estimates. In this case, the tool assumes default toggle rates and static probabilites, and uses heuristic algorithms, to compute probable power. A major drawback of vectorless power estimation is that the algorithm cannot determine glitching activity, i.e., multiple logic transitions in the same clock cycle. This can result in a significant underestimation of device dynamic power, especially in designs with long logic paths which have higher probability of glitching.

A more accurate power estimate can be achieved using a post-implementation power model. This model is applied to a post-implementation timing simulation (executed in Xilinx iSim, Vivado Simulator, or Mentor Graphics QuestaSim), along with relevant test vectors, to measure actual toggle rates and static probabilities. These events are captured in text files, such as a switching activity information file (.saif), or value change dump (.vcd). The user is likewise responsible for determining very accurate environmental conditions, such as ambient temperature, airflow, and exact configuration of an FPGA board in a chassis. Although more accurate, construction of a test bench capable of running long, representative test vectors, generation of .saif or .vcd files, and verification of proper toggle and static probability rates in the power analysis software, require a significant level of operator skill and time. Additionally, manufacturing process variations on an individual board can have a large impact – sometimes affecting the static power consumption by greater than a factor of two.

Since we are already instantiating a live version of each authenticated cipher in actual hardware, using long test vectors fully representative of authenticated cipher operations, it is desirable to determine power and energy consumption on actual hardware. We adapt the FOBOS architecture to measure power consumed by the Spartan-6 1.2V bus, e.g., $V_{CCINT}$, by measuring current through a 1Ω shunt resistor. Our current is amplified by the TI INA225 amplifier, rendered as a voltage in an oscilloscope, and offloaded to an attached PC for post-run power computation.

Power measurements are recorded at discrete time intervals corresponding to sample rate. Our current FOBOS installation computes about 20,000 samples per trace. Between 10 and 100 traces (using various test vectors of up to 2000 bytes each) are used to generate power traces. The power measurements contain a combination of static and dynamic power at each sample. For our typical authenticated cipher design, $V_{CCINT}$ accounts for more than 95% of the dynamic power, but only about 20% of the total static power (the other static power consumers are the 2.5V $V_{CCAUX}$ and 3.3V $V_{CCO}$). Therefore, our power measurement underestimates actual device usage at lower frequencies, but improves in accuracy with increasing frequency (i.e., as a larger share of power becomes dynamic). Our methodology also accurately captures the relative increase in dynamic power between unprotected and protected versions. Additionally, the FIFOs in the FOBOS DUT wrapper (instantiated as BRAMs), as well as ancillary logic necessary to feed the authenticated ciphers and extract results, consume some of the total measured power, which results in additional error.

During post-analysis, mean power ($P_{mean}$) is computed by averaging instantaneous power measurements over the entire time domain, while maximum power ($P_{max}$) is estimated by sampling the highest peaks during each trace. Energy-per-bit (nJ/bit) is then estimated as $P_{mean}(mJ/s)/TP(Mbps)$, where TP (throughput) is the throughput of an authenticated encryption of a long message. Note that estimating TP based on a long message tends to

negate "short-message" abnormalities, such as authenticated data processing, key, or state variable initializations.

# 4 Protection of Authenticated Ciphers against DPA

The authenticated ciphers introduced in subsection 2.1 are protected against 1st order Differential Power Analysis (DPA) using a maximum of a 3-share threshold implementation (TI) (as described in subsection 2.3), where the uniformity property is achieved using internal randomness provided by pseudo-random number generators (PRNG) constructed using linear feedback shift registers (LFSRs). Protection strategies for individual ciphers are outlined below.

## 4.1 Threshold Implementation (TI) of ACORN

ACORN is the only stream authenticated cipher evaluated in this research. ACORN can be implemented serially, or in $n$-bits of output generated in parallel. We choose the very lightweight 8-bit architecture (ACORN-8) available at [16]. Although the 8-bit internal datapath can possibly be protected using a basic iterative architecture, we choose to execute the state update in two clock cycles instead of one, in order to distribute the non-linearity across two clock cycles. We instantiate ten 8-bit hybrid 2- / 3- share TI-protected and functions, each of which consumes 16 random reshare, and 8 random refresh bits, to maintain the TI uniformity during each call. Amortized over two clock cycles, this results in an average of 120 random bits per clock cycle, which are provided by a pseudo random number generator (PRNG). An abbreviated representation of the ACORN state update is shown in Fig. 5.

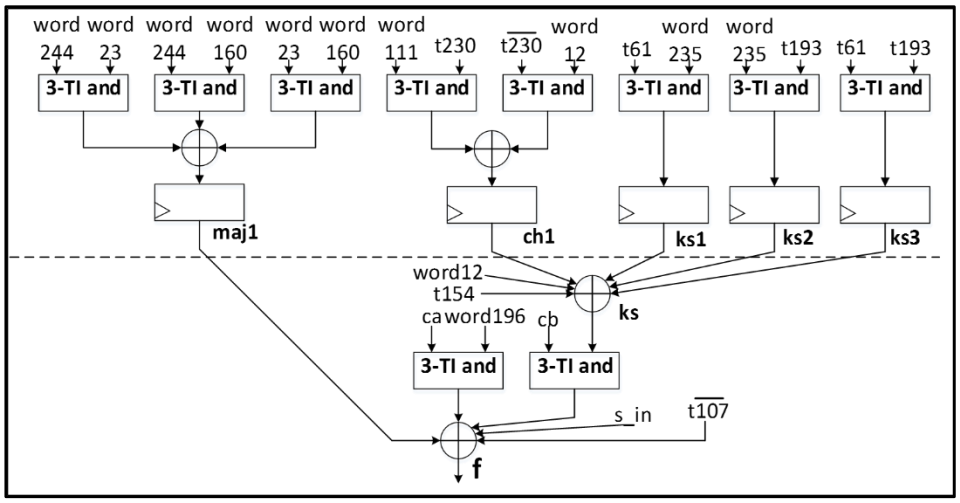


Fig. 5. Simplified ACORN two-cycle state update. Dashed line indicates boundary between clock cycles. All signals and buses represent two TI-shares. Signals are defined in source code [16], and derived from [9]. All bus widths are 8 bits.

## 4.2 Hybrid 2- / 3- share TI-protected ASCON

ASCON uses a custom substitution-permutation (SPN) transformation based on duplex sponge modes like Monkey Duplex [10]. The ASCON implementation at [15], with 64-bit block size and internal datapath, and basic-iterative architecture, is not ideal for protection against DPA. In order to minimize resources required for a 3-share 64-bit TI-protected `and` module, reduce required random refreshing and resharing bits, and reduce vulnerability due to energy and information leakage, we implement a hybrid 2- / 3- share TI-protected ASCON which executes one round in seven clock cycles. We use the bitslice S-Box discussed in [10], and instantiate only one hybrid 2- / 3- share 64-bit TI-protected `and` function, which uses 192 random bits per clock cycle – 128 bits for resharing (from two to

three shares), and 64 bits to satisfy the TI uniformity property. Round constant addition occurs on the 320-bit state in Stage 0; S-Box operations occur in Stages 0 through 6; and linear permutations occur in Stage 6.

The randomness is provided by a 192-bit PRNG, which also performs pre-whitening during state initialization to begin round computations with an average Hamming Weight (HW) of 0.5-per-bit. We modify the unprotected version of ASCON at [15] to use the same seven-cycle architecture to facilitate fair benchmarking. The modified ASCON S-Box is in Fig. 6.

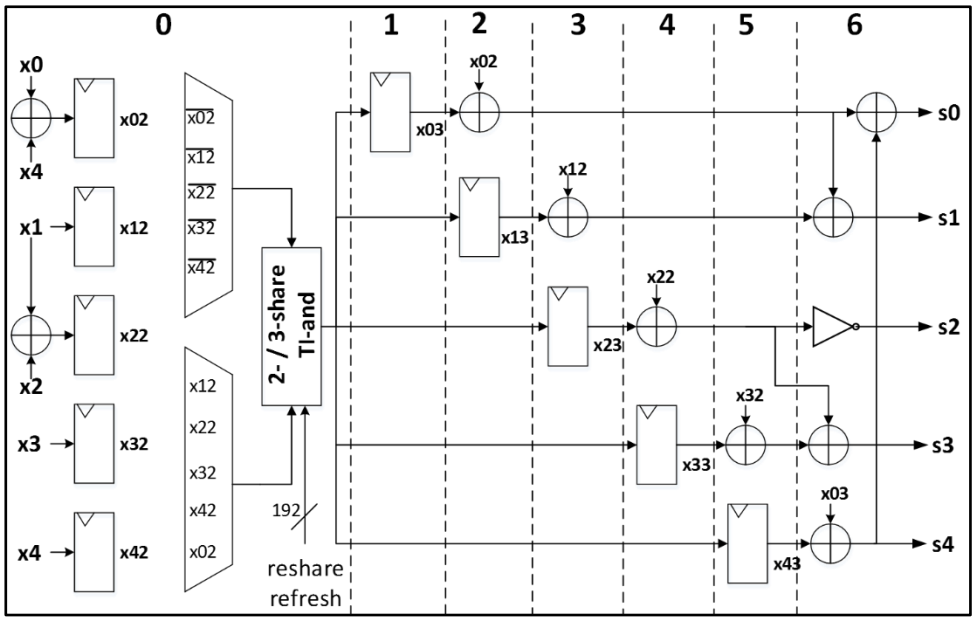


Fig. 6. 7-cycle ASCON S-Box. Stages are separated by dashed lines, and numbered across the top. All bus widths are 64-bits, unless indicated.

### 4.3 TI protection of AES in AES-GCM, CLOC, SILC, and JAMBU

TI-protected versions of AES are documented in [22, 31, 32, 36]. We improve upon on the hybrid 2- /3- share 5-stage pipelined version in [22] by upgrading the pipeline with TI-protection for round keys generated on the fly. Our TI-protected AES uses an S-Box using combinational logic, as described in [37, 38]. Using the method of Tower Fields, where inversions in $GF(2^8)$ are represented as operations in $GF(2^4)$, which are in turn represented in $GF(2^2)$, field multiplications and inversions in low-degree non-linear representations become feasible.

Since the $GF(2^8)$ inverter portion of a TI-protected S-box is costly, we instantiate only one 3-share TI-protected 8-bit inverter. Outside the inverter, we adopt a method described in [31] to employ a hybrid 2-/3-share TI approach, where linear calculations (such as round key addition, column multiplications, basis conversions, affine transformations, etc.) are conducted on only two shares to save resources.

Our resulting protected design has a 5-stage pipeline, where one S-Box operation commences every clock cycle. A 128-bit round completes every 20 cycles, and a 128-bit block encryption executes in 205 clock cycles (including five cycles to prime the pipeline). The design uses 16 bits of fresh randomness for resharing from two to three shares, and two fresh remasking bits per $GF(2^2)$ multiplier and multiplier-scalar instance, resulting in a total of 40 random bits required for each S-Box (i.e., per clock cycle). The required refresh randomness is supplied by a PRNG integrated into the AES core. The S-Box computation, distributed over five pipeline stages, is shown in Fig. 7.

The authenticated ciphers at [15] using AES as a cryptographic primitive (i.e., AES-GCM, CLOC, SILC, and JAMBU) are optimized for high-speed operations, and use a full-width AES core which executes a 128-bit block encryption in 10 clock cycles. However, it is not feasible to build a full-width TI-protected AES with basic iterative architecture, due to 1) quadratic increase in resources for TI-protection, 2) large number of random refresh bits required, and 3) probability of increased vulnerability to SCA due to long paths of combinational logic along which glitches can occur. Therefore, in order to facilitate a relevant benchmarking comparison between unprotected and protected ciphers, we replace the full-width AES with an unprotected version of our 8-bit, 5-stage pipelined AES in the unprotected implementations of AES-GCM, CLOC, SILC, and JAMBU.

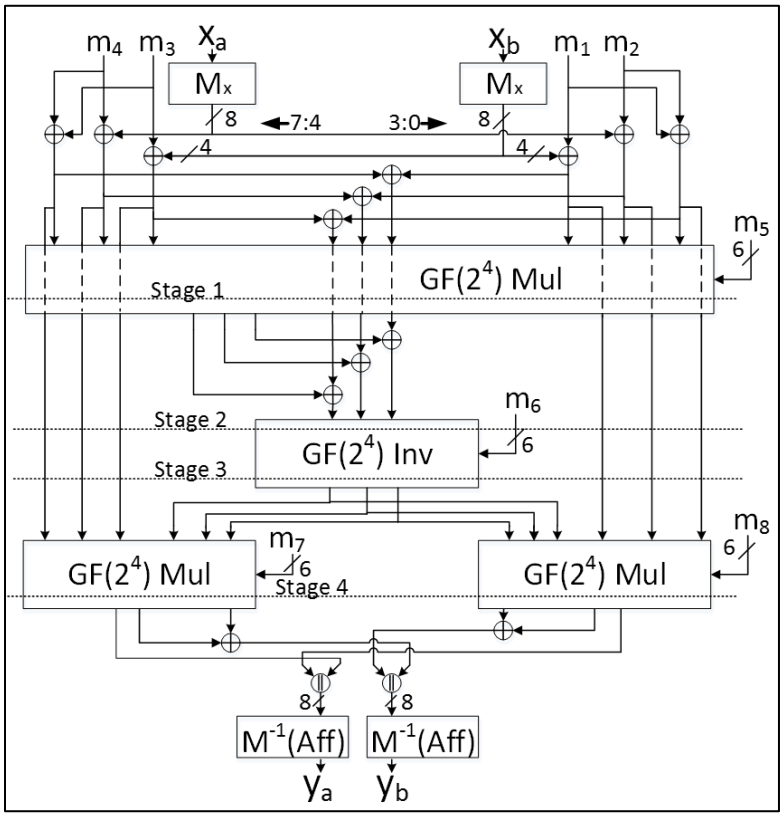


Fig. 7. 8-bit 5-stage pipelined TI-protected S-Box. Dotted lines denote stage boundaries; dashed lines indicate pass-thru, i.e., no interaction with component. Mx represent change to normal basis, and $M^{-1}$ are combined affine transformation and change to standard basis. $m_i$ indicate random refresh bits, with bus widths as indicated. Adapted from [22].

### 4.4 TI protection of the GF($2^{128}$) Multiplier in AES-GCM

AES-GCM is different than all of the other authenticated ciphers in this study, in that it has a significant non-linear operation outside of the cryptographic primitive, namely, multiplication in GF($2^{128}$) modulo the polynomial $x^{128} + x^7 + x^2 + x + 1$ ($P(x)$). Since the TI-protection of this multiplier is bound to be costly, an interesting question is whether or not this operation should be protected to prevent vulnerability to DPA. According to [14], the secret key itself is never used in the multiplier. Rather, combinations of plaintext, ciphertext, authenticated data, and block length information are processed by the multiplier.

There are known weaknesses associated with AES-GCM. One example is the Ferguson Observation, where it is possible to create a tag linearly dependent on the hash key $K_H$, since multiplications by $2^n$ are linear [39]. Another example is a 1st order DPA attack on AES in the counter mode [40]. Additionally, Belaid et al concluded that attacking a multiplier in AES-GCM could provide knowledge of the AES secret key $K_S$, and determined that a

designer should mask the multiplier to protect against Hamming Weight (HW) leakage in registered values [36]. The AES-GCM implementation documented in [41] notes these potential vulnerabilities, and provides a TI-protected multiplier.

Even if we did not protect the multiplier, we would be required to combine the two shares of sensitive data leaving the AES core, perform the multiplication, and potentially reshare the data into two shares for subsequent operations, which would incur additional logic and randomness requirements. Additionally, an unprotected multiplier would not satisfy our verification methodology, since it would certainly fail our t-test, regardless of vulnerabilities to subsequent key recovery attacks.

Based on the above logic, we develop a low-cost 3-share TI-protected multiplier, which computes $a \cdot b \bmod P(x)$, where $a$ and $b$ are 128-bit operands. The multiplier executes a two-operand multiplication in 128 clock cycles. This does not affect the overall throughput for large messages, since it is still less than the 205 clock cycles required for an AES block encryption. The multiplier is low-cost in area, since multiplication in each clock cycle is only a 128-by-1 bit multiplication, followed by parallel exclusive-OR gates to provide reduction modulo $P(x)$. The randomness for resharing from two to three shares in the multiplier is recycled from a shift register containing randomness at the input stages of the AES core provided by the PRNG. The use of recycled randomness presents a potential vulnerability for higher orders of DPA, but is practically uncorrelated to our multiplicands after sensitive data is permutated by an entire AES block encryption. The 3-share TI-protected multiplier is shown in Fig. 8.

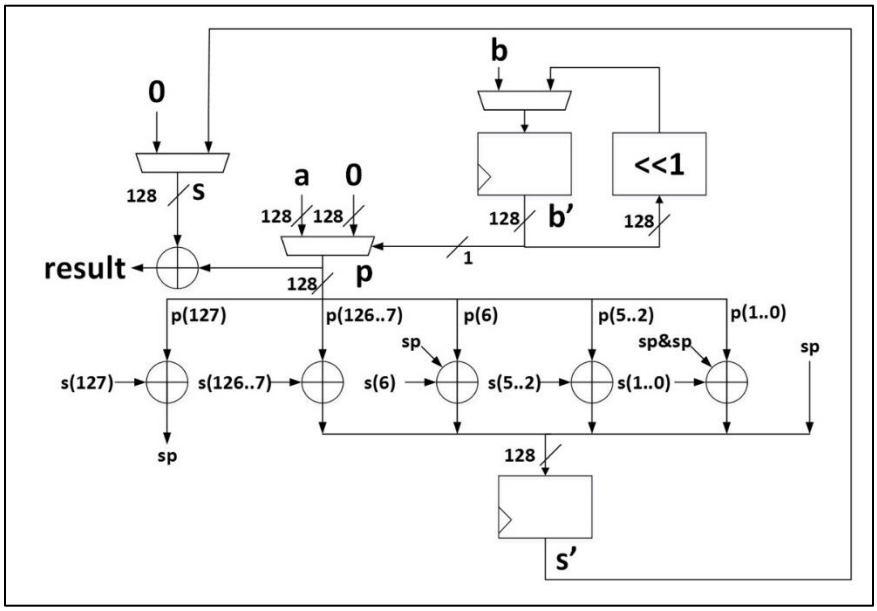


Fig. 8. 3-share TI-protected GF($2^{128}$) multiplier used in AES-GCM. All signals are triplicated inside the multiplier for three-share computation.

### 4.5 TI protection of SIMON in JAMBU-SIMON

The authors of JAMBU-SIMON published an implementation with unrolled architecture, i.e., four rounds per clock cycle, as their CAESAR Round 3 hardware submission [18]. Although uncertain of our chances of success, we choose to directly apply 3-share TI protection to the unrolled JAMBU-SIMON. A 3-share threshold implementation (TI) of SIMON is easily achieved using the methodology described in [22, 24, 27]. SIMON uses only a single 2-input 48-bit `and` gate to achieve non-linearity. Therefore, a 3-share TI of this quadratic equation is achieved without requiring any cascading or composite functions.

The TI non-completeness and correctness properties are satisfied, as each share lacks at least one of the component shares in its calculation. The uniformity property is satisfied in this case by considering the key shares, included in each TI-share calculation, as a source of randomness [24]. Therefore, no mask refreshing is required in SIMON 3-share TI, which leads to a generally efficient TI-protected implementation. A sample SIMON round is shown in Fig. 9.

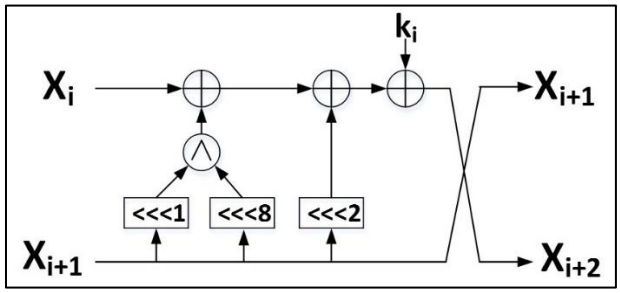


Fig. 9. One round of SIMON 96/96 primitive used in JAMBU-SIMON. All bus widths are 48 bits.

### 4.6 TI protection of PRESENT and LED in SILC

PRESENT and LED use the same 4-bit S-Box. The S-Box is a cubic function, but can be decomposed into two quadratic functions, which can be computed using 3-TI shares [42, 43]. As discussed in [42, 43], the output of the composite functions is a permutation on the input bits, meaning that the TI uniformity property is satisfied without additional random refresh bits. We choose to protect all 16 S-Boxes in parallel (plus associated S-Boxes for round key updates) in the full-width datapath (i.e., 64 bits), basic iterative architecture, as implemented in [17]. We implement strict 3-share TI-protection, and thus do not require any random reshare bits. The composite 3-share TI-protected S-Box used in PRESENT and LED is shown in Fig. 10.

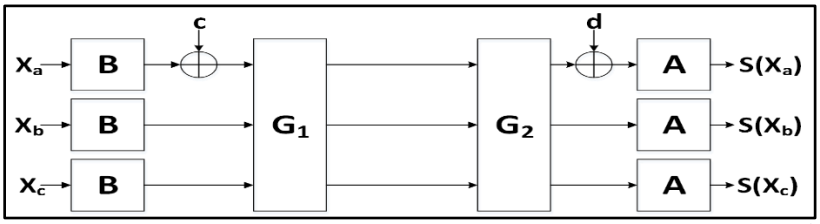


Fig. 10 3-share TI-protected S-Box used in authenticated ciphers with PRESENT and LED, adapted from [22, 42, 43]. A and B are matrix multiplications; c and d are constants; G is a quadratic composite function. All bus widths are 4 bits.

### 4.7 TI protection of TWINE in CLOC-TWINE

Like PRESENT and LED, TWINE also uses a 4-bit S-Box of algebraic degree 3 which can be decomposed into two quadratic functions. We leverage Fermat's Little Theorem, as described in [22, 44] to compute $x^{14} \equiv x^{-1}$ in GF($2^4$), which decomposes into two non-linear multipliers, with several low-cost linear squares, as shown in Fig. 11.

The outputs of the multipliers, however, are not permutations on the input; they do not satisfy the TI uniformity property. Therefore, we use two bits of refresh randomness per S-Box per clock cycle, for a total of 20 random bits per clock cycle, including four bits for the S-Boxes for round key updates. MATLAB Monte Carlo simulations demonstrated that single-bit mask refreshing (for each 4-bit multiplier) achieves an average probability of an

output '1' of 0.499 for the four output bits (given equally likely '0' or '1' at input bits), with a minimum single bit probability of 0.498.

Like PRESENT and LED, we seek to implement a strict 3-share TI-protected implementation on the full-width (64-bit) datapath, using the basic iterative architecture, as implemented in [17].

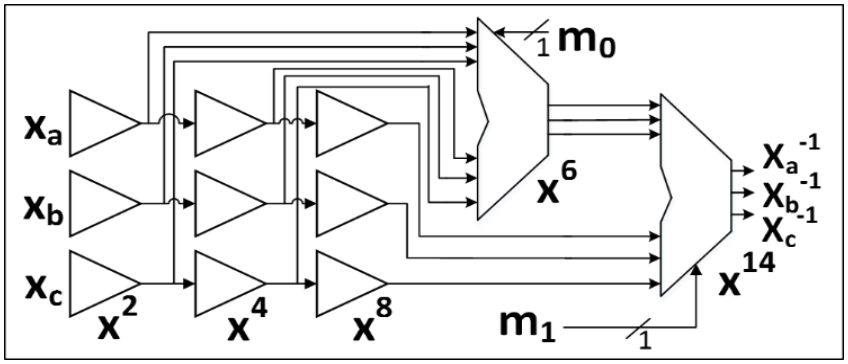


Fig. 11 3-share TI GF($2^4$) inverter used in TWINE. $x^2$, $x^4$, $x^8$ (produced by squares), and $x^6$ (where $x^6 = x^2 \cdot x^4$) are intermediate products; $x^{14}$ (where $x^{14} = x^6 \cdot x^8$) is final product; All bus widths are 4 bits, except for random bits $m_0$ and $m_1$, which are single bits. Adapted from [22].

### 4.8 TI protection of Ketje Jr.

The authors of the Ketje Jr. implementation at [19] use a basic-iterative architecture with full-width datapath, presumably to maximize TP/A ratio. This means that a large number of non-linear operations will occur in each clock cycle, and increases the chances of providing inadequate protection against DPA due to propagation of glitches. Additionally, the protection cost is potentially large. However, Ketje Jr. is the smallest of available CAESAR Round 3 Ketje specifications, and has a state of only 200 bits, which increases our chances of success.

TI protection of Ketje is relatively straight-forward, since Ketje uses the Keccak-$p^*$ transformation (adapted from the Keccak-p in SHA-3). Only one transformation ($\chi$) is non-linear, and protection is provided by a 3-share TI-protected AND module. We implement a hybrid 2- / 3-share TI-protection, using two shares outside the $\chi$ transformation, resharing to three shares in $\chi$, and recombining to two shares for the remainder of the round. We use 200 bits of resharing randomness per clock cycle, which is provided by an integrated PRNG. The protected Keccak-$p^*$is shown in Fig. 12.

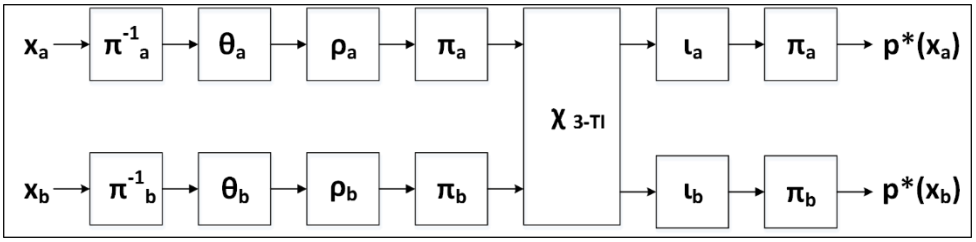


Fig. 12 Hybrid 2- /3-share TI-protected Keccak-$p^*$ transformation, used in Ketje Jr.

### 4.9 Summary of Authenticated Ciphers

The characteristics of the authenticated ciphers investigated in this research, including the architectural choices used to achieve protection against 1st order DPA, are summarized in Table 2.

Table 2. Characteristics of Authenticated Ciphers Examined in this Research

| Cipher | Architecture | Rnds | Cycles per Block | Formula for Throughput | Random bits per clock cycle |
|---|---|---|---|---|---|
| AES-GCM | 8-bit 5-stage Pipl. AES, 128-cycle GF multiplier | 10 | 218 | $(128/218)*f_{clk}$ | 40 |
| ACORN | 8-bit 2-cycle folded | - | 2[1] | $4 * f_{clk}$ | 120 |
| ASCON | 64-bit 7-cycle folded | 7 | 49 | $(64/49) * f_{clk}$ | 192 |
| CLOC-AES | 8-bit 5-stage Pipl. AES (two cores) | 10 | 206 | $(128/206)*f_{clk}$ | 40 |
| CLOC-TWINE | 64-bit basic-iterative | 36 | 70 | $(64/70) *f_{clk}$ | 20 |
| SILC-AES | 8-bit 5-stage Pipl. AES (two cores) | 10 | 205 | $(128/205)*f_{clk}$ | 40 |
| SILC-PRESENT | 64-bit basic-iterative | 31 | 64 | $(64/64) *f_{clk}$ | 0 |
| SILC-LED | 64-bit basic-iterative | 48 | 98 | $(64/98) *f_{clk}$ | 0 |
| JAMBU-AES | 8-bit 5-stage Pipl. AES | 10 | 205 | $(64/205)*f_{clk}$ | 40 |
| JAMBU-SIMON | 48-bit Unrolled x4 | 52 | 13 | $(48/13) *f_{clk}$ | 0 |
| Ketje Jr. | 32-bit basic-iterative | 2 | 2 | $(32/2) *f_{clk}$ | 200 |

"Formula for Throughput" is for a long message consisting of Message (i.e., plaintext or ciphertext), where $f_{clk}$ is clock frequency. "Rounds" abbreviated "Rnds." "Pipelined" abbreviated "Pipl." Note 1 – 2 cycles per block after state initialization.

# 5 Results

## 5.1 Power analysis of unprotected authenticated ciphers

We use the above FOBOS t-test leakage detection methodology to measure the resistance of AES-GCM, ACORN, ASCON, CLOC (AES, TWINE), SILC (AES, PRESENT, LED), JAMBU (AES, SIMON), and Ketje Jr. to DPA. We perform 2000 "fixed-versus-random" high fidelity traces (i.e., between 16,000 and 20,000 samples per trace), using aeadtvgen.py-formatted test vectors, consisting of between four and eight combinations of authenticated encryption and decryption. The t-tests are performed on the Nexys-3 victim board, and instantiated in the Spartan 6 FPGA (xc6slx16csg324-3). For t-tests, the ciphers are clocked at 781 KHz, in order to minimize capacitive and inductive effects and present a cleaner power signature. The results, shown in Fig. 13, indicate significant leakage in all cipher implementations. The results are as expected for unprotected implementations.

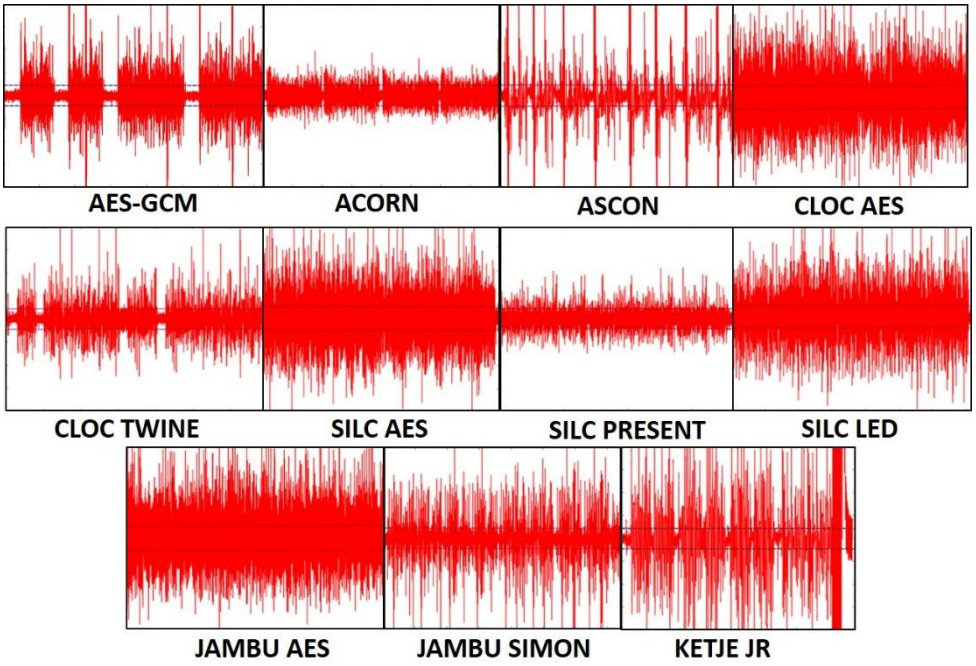


Fig. 13. Results of t-test on unprotected ciphers. Time domain (samples) are on the x-axis, t-values are on the y-axis. Horizontal lines denote t = ± 4.5.

### 5.2 First attempt at protection of AES-JAMBU

We attempt to provide an implementation of AES-JAMBU, resistant to 1st order DPA. The JAMBU layer above the AES core consists of linear operations, and is separated into two shares, using random data supplied through the `rdi` interface. We note that padding, a required operation for JAMBU (and most authenticated ciphers) is not a linear operation. However, padding is performed in PreProcessor (and not in CipherCore, shown in Fig. 3) in this implementation, and will be addressed later.

When share separation is performed in CipherCore, the amount of initial randomness required is equal to $\#rnd\ bits = \#bits\ bdi + \#bits\ key$. Note that this does not include refresh masking required during cipher operation (described in [24 –26]), which is provided by an integrated PRNG. The arrival timing of random data is also important; it must be received prior to the separation of incoming data in `bdi` and `key` (defined in [21]). In our first iteration, we achieve synchronization through modification of the AES-JAMBU controller. However, this is a suboptimal approach which is not standardized for multiple authenticated ciphers, and defeats our goal of protection and analysis of a large number of ciphers (a remedy will be discussed subsequently).

We perform a t-test on the AES-JAMBU as modified above. The results (shown in Fig. 14(a)) indicate that leakage is significantly reduced, yet still present at numerous discrete points throughout the time domain. We align the time domain of the t-test plot with key events in Xilinx iSim, and note that the location and frequency of t-test spikes where $|t| > 4.5$ appears to match important I/O events (shown in Fig. 14(b)). We hypothesize that leakage occurs because PreProcessor and PostProcessor register unmasked data (both entering and leaving the cipher), which allows the t-test (or a potential attacker) to correlate sensitive data. In other words, the Development Package I/O processors themselves (as implemented in [7]) leak information.

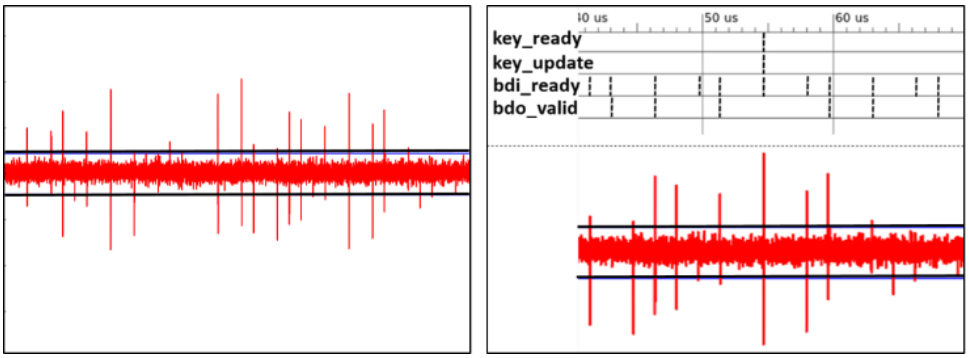


Fig. 14(a) (left) T-test result for protected version of AES-JAMBU CipherCore; 14(b) (right) Time-domain alignment of leakage spikes with key I/O signal events in Xilinx iSim simulation of the same cipher.

### 5.3 Second attempt at protection of AES-JAMBU

To validate our hypothesis, we produce a 2- / 3- share TI-protected version of Pre- and Post-Processor, by modifying the designs at [7]. This ensures that unmasked sensitive data is never registered during cipher operation. As discussed, padding is a non-linear operation, and is performed using a 3-share TI-protected OR in PreProcessor.

Since randomness is required in PreProcessor, we streamline the ingestion of randomness by designing a separate `rdi` PreProcessor, an approach which is generic and applicable to all protected authenticated ciphers using the Development Package at [7]. In this approach, the amount of randomness required in `rdi` is $\#rnd\ bits = (\#bits\ public\ data + \#bits\ key\ data) \times (d - 1)$, where $d$ is the number of TI shares. There is one additional potential source of leakage – the FOBOS DUT victim-board test wrapper itself. Since sensitive data is occasionally registered in the wrapper, and we cannot isolate the wrapper during testing, we build a 2-share TI-protected wrapper (using a separate PRNG integrated into the FOBOS DUT wrapper). The successful t-test of JAMBU-AES using the modified AEAD, including modified Pre- and Post-Processor, and `rdi` interface, is shown in Fig. 15. The updated and augmented CAESAR HW interface, with modified I/O processors, is shown in Fig. 16.

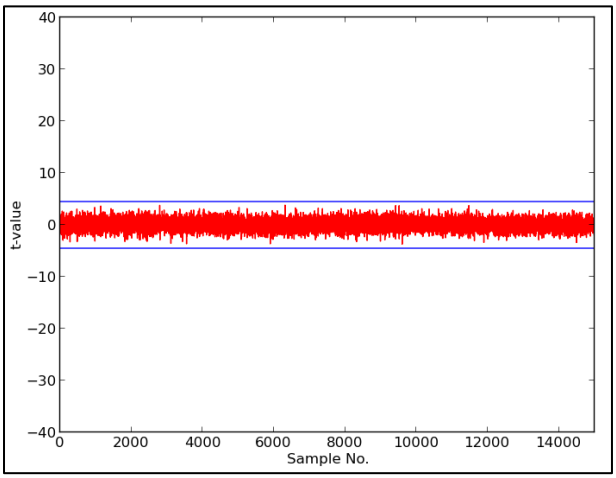


Fig. 15 Successful t-test of JAMBU-AES after modification of AEAD, including Pre- and Post-Processors.

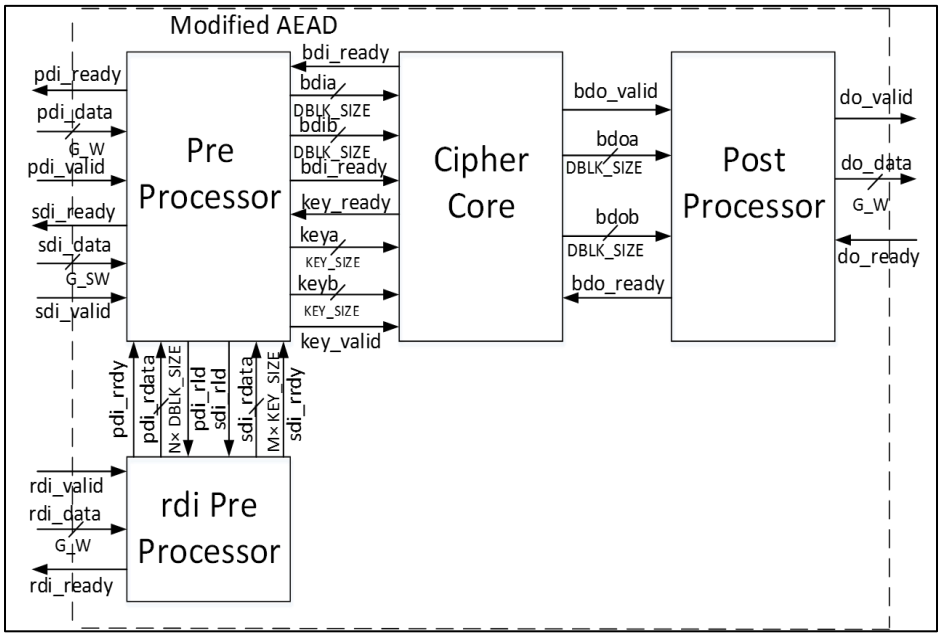


Fig. 16. External interface of the modified authenticated cipher module (Modified AEAD), and the internal top-level block diagram of Modified AEAD, augmented with rdi PreProcessor

### 5.4 Protection of Remaining Authenticated Ciphers

All remaining authenticated ciphers are updated using their respective cryptographic primitives (as described in "Methodology"), and instantiated in the Modified AEAD module, as described above. The authenticated cipher layers above the primitive in CipherCore are protected using either 2-share TI (e.g., AES-GCM, ACORN, ASCON, CLOC-AES, SILC-AES, JAMBU-AES and Ketje Jr.) or 3-share TI (e.g., CLOC-TWINE, SILC-PRESENT, SILC-LED, JAMBU-SIMON). TI protection of cipher functionality within CipherCore is relatively straightforward, except that one must take care to account for occasional non-linear operations, such as padding, and ensure that derived control functions do not leak information.

The t-test leakage detection methodology shows that the AES-GCM, ACORN, ASCON, SILC, JAMBU, and Ketje Jr. ciphers pass the t-test, and are resistant to 1st order DPA. The results of protected ciphers are shown in Fig. 17.

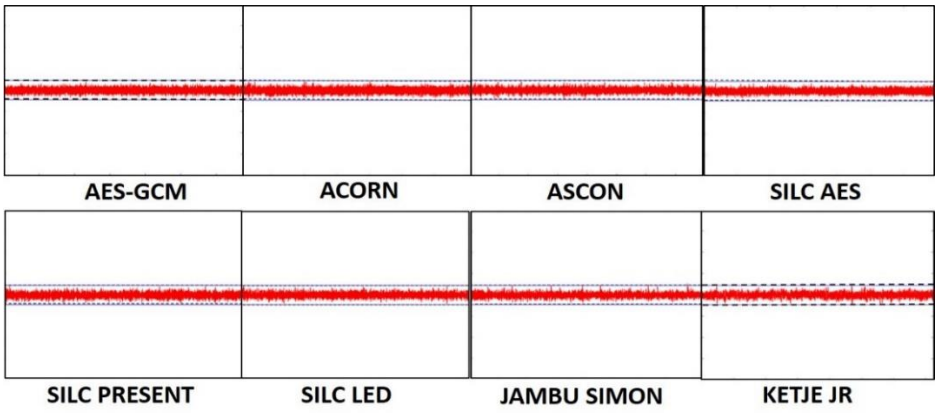


Fig.17. Results of t-test on AES-GCM, ACORN, ASCON, SILC, JAMBU (SIMON), and KETJE authenticated ciphers, protected against 1st order DPA. The dashed lines denote t = ± 4.5.

### 5.5 Limitation of the t-test leakage detection methodology for authenticated ciphers

In our t-test leakage detection methodology, failures of t-tests do not always indicate vulnerability to DPA. One case in which a t-test could indicate "leakage," but not necessarily vulnerability to DPA, occurs in the case of the CLOC authenticated cipher. The CLOC specification contains a data- dependent decision condition. As shown in Alg. 1, a tweak ("h") is performed based on the most-significant-bit (msb) of the first word of associated data $A[1]$. Whether the algorithm is implemented in hardware or software, some decision mechanism must choose whether or not to use the $h(S_H[1])$ result. In the case of the CLOC-AES hardware implementation, this decision is made in a Finite-State Machine (FSM) and communicated to a multiplexer in the datapath.

Algorithm 1. Excerpt from CLOC $HASH_K$(N,A) Algorithm [11]

| |
|---|
| 1. $if\ msb_1\big(ozp(A[1])\big) = 1\ then$ |
| 2. $S_H[1] \leftarrow h(S_H[1])$ |

We hypothesize that the t-test will detect this data-dependent decision condition, regardless of our TI-protected implementation. To test this hypothesis, we first conduct a t-test of our protected implementation of CLOC-AES, with an unconstrained fixed-versus random test vector. The results, shown in Fig. 18, show that the t-test generally passes, but unambiguously fails at discrete points.

Next, we generate two sets of alternative test vectors, using aeadtvgen.py and the FOBOS fixed-versus-random test vector generator, where $msb(A[1])$ is constrained to be either 0 or 1 (labeled $AD_0$ or $AD_1$, respectively), and re-run the t-tests. The results in Fig. 18 show fully passing t-tests for both test vectors, which validates our hypothesis that the t-test is able to detect this data-dependent decision.

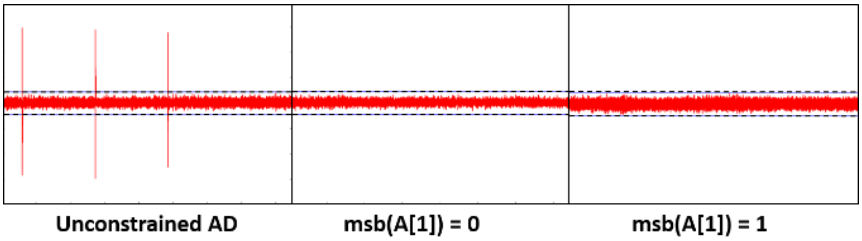


Fig. 18. Results of t-tests on protected implementation of CLOC-AES, with unconstrained test vectors on left, and test vectors $AD_0$ and $AD_1$ in center, and right, respectively. Time domain (samples) are on the x-axis, t-values are on the y-axis. The horizontal lines denote t = ± 4.5.

Although this data-dependent conditional decision in CLOC inevitably fails our t-test, it does not expose CLOC-AES to DPA vulnerability, since the conditional decision occurs only on publicly-available *AD*, and does not depend on secret key. Our manipulation of t-test vectors to eliminate a failed t-test based on $msb(A[1])$ shows that we have achieved a protected version of CLOC-AES resistant to 1st order DPA. An analogous series of t-tests, with similar results, is illustrated for the CLOC-TWINE cipher in Fig. 19.

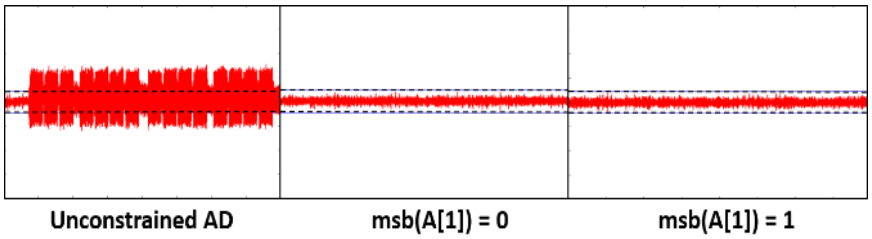


Fig. 19. Results of t-tests on protected implementation of CLOC-TWINE, with unconstrained test vectors on left, and test vectors $AD_0$ and $AD_1$ in center, and right, respectively. Time domain (samples) are on the x-axis, t-values are on the y-axis. The horizontal lines denote t = ± 4.5.

### 5.6 Benchmarking of unprotected and protected ciphers

We implement the unprotected and protected versions of the 11 ciphers using Xilinx ISE on the Spartan 6 FPGA. The results are further optimized for throughput-to-area ratio using the ATHENa optimization tool [45]. During optimization, we prohibit Block RAM generation, in order to ensure that area (LUTs and slices) are directly comparable. The results, in terms of LUTs, slices, frequency, throughput, and throughput-to-area (TP/A) ratio, are shown in Table 3 and displayed in Fig. 20. Note that protected implementations include any required PRNG in Place and Route (P&R) results.

Using the FOBOS architecture, we estimate mean power ($P_{mean}$) and maximum power ($P_{max}$) (mW) for all ciphers at 10 MHz, where the victim board is supplied by an external frequency generator. Energy-per-bit (nJ/bit) is computed as $P_{mean}(mJ/s)/TP(Mbps)$. Results are shown in Table 4, and displayed in Figs. 21 and 22.

Table 3. Optimized Implementation Results of Authenticated Ciphers in Spartan-6 FPGA

| Cipher | Area | | Ratio | Freq | TP | Ratio | TP/A | Ratio |
|---|---|---|---|---|---|---|---|---|
| | LUT | Slices | Pr/UnPr (LUT) | MHz | Mbps | UnPr/Pr (Mbps) | Mbps/LUT | UnPr/Pr (Mbps/LUT) |
| Unprotected (UnPr) | | | | | | | | |
| AES-GCM | 1947 | 688 | - | 176.0 | 103.4 | - | 0.0531 | - |
| ACORN | 549 | 269 | - | 226.6 | 906.2 | - | 1.6507 | - |
| ASCON | 2048 | 755 | - | 195.5 | 255.4 | - | 0.1247 | - |
| CLOC-AES | 2496 | 1108 | - | 150.0 | 93.2 | - | 0.0373 | - |
| CLOC-TWINE | 1536 | 485 | - | 171.2 | 156.5 | - | 0.1019 | - |
| SILC-AES | 1975 | 755 | - | 163.0 | 101.7 | - | 0.0515 | - |
| SILC-PRESENT | 2057 | 610 | - | 238.8 | 238.8 | - | 0.1161 | - |
| SILC-LED | 1990 | 699 | - | 203.4 | 132.8 | - | 0.0667 | - |
| JAMBU-AES | 1073 | 527 | - | 163.1 | 50.9 | - | 0.0475 | - |
| JAMBU-SIMON | 1105 | 311 | - | 137.9 | 509.3 | - | 0.4609 | - |
| KETJE JR | 1242 | 363 | - | 96.9 | 1550.4 | - | 1.2483 | - |
| Protected (Pr) | | | | | | | | |
| AES-GCM | 4828 | 1870 | 2.48 | 116.8 | 68.57 | 1.51 | 0.0142 | 3.74 |
| ACORN | 2732 | 1032 | 4.98 | 142.7 | 570.6 | 1.59 | 0.2089 | 7.90 |
| ASCON | 6364 | 2062 | 3.11 | 103.1 | 134.6 | 1.90 | 0.0212 | 5.89 |
| CLOC-AES | 5900 | 2157 | 2.36 | 104.2 | 64.7 | 1.44 | 0.0110 | 3.40 |
| CLOC-TWINE | 6467 | 2073 | 4.21 | 70.7 | 64.7 | 2.42 | 0.0100 | 10.19 |
| SILC-AES | 4865 | 1899 | 2.46 | 102.8 | 64.2 | 1.59 | 0.0132 | 3.91 |
| SILC_PRESENT | 4624 | 1638 | 2.25 | 116.6 | 116.6 | 2.05 | 0.0252 | 4.60 |
| SILC-LED | 4780 | 1550 | 2.40 | 92.0 | 60.1 | 2.21 | 0.0126 | 5.31 |
| JAMBU-AES | 2869 | 1105 | 2.67 | 122.4 | 38.2 | 1.33 | 0.0133 | 3.56 |
| JAMBU-SIMON | 3140 | 1243 | 2.84 | 58.7 | 216.7 | 2.35 | 0.0690 | 6.67 |
| KETJE JR | 4800 | 1879 | 3.86 | 59.6 | 954.0 | 1.63 | 0.1987 | 6.28 |

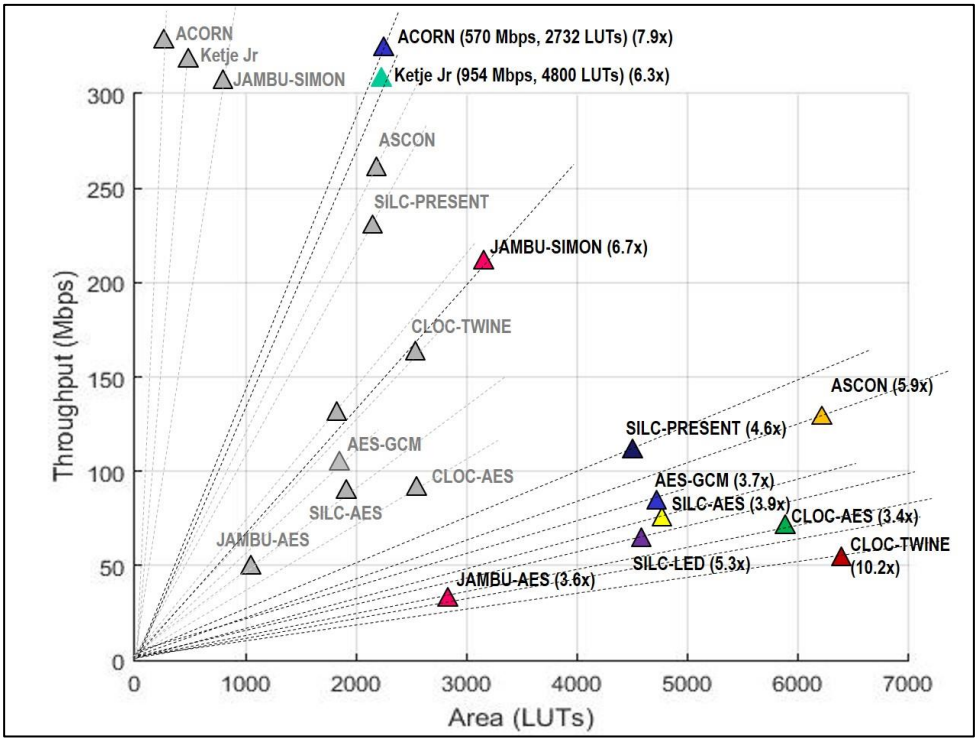


Fig. 20. Throughput, area (in LUTs), and throughput-to-area (TP/A) ratios of unprotected and protected authenticated ciphers. Unprotected versions are shown in gray triangles, while protected versions are depicted with darker triangles. The relative increase in TP/A ratio is shown next to protected versions.

Table 4. Power and Energy-per-bit Measured on Spartan-6 FPGA at 10 MHz

| | Power (mW) | | Ratio | Pmax-Pmean | Energy | Ratio |
|---|---|---|---|---|---|---|
| Cipher | Pmean | Pmax | Pr/UnPr (mW) | % diff | nJ/bit | Pr/UnPr (nJ/bit) |
| Unprotected (UnPr) | | | | | | |
| AES-GCM | 10.3 | 11.5 | - | 11.7 | 1.754 | - |
| ACORN | 7.8 | 8.6 | - | 9.9 | 0.195 | - |
| ASCON | 10.5 | 11.5 | - | 8.8 | 0.805 | - |
| CLOC-AES | 12.4 | 14.0 | - | 12.9 | 1.996 | - |
| CLOC-TWINE | 10.3 | 11.6 | - | 12.5 | 1.129 | - |
| SILC-AES | 10.6 | 13.1 | - | 23.6 | 1.698 | - |
| SILC-PRESENT | 9.7 | 10.7 | - | 9.8 | 0.972 | - |
| SILC-LED | 10.9 | 12.0 | - | 10.1 | 1.666 | - |
| JAMBU-AES | 9.4 | 10.0 | - | 6.7 | 3.001 | - |
| JAMBU-SIMON | 19.7 | 21.0 | - | 6.6 | 0.534 | - |
| KETJE JR | 22.0 | 26.5 | - | 20.5 | 0.138 | - |
| Protected (Pr) | | | | | | |
| AES-GCM | 23.9 | 28.1 | 2.32 | 17.6 | 4.070 | 2.32 |
| ACORN | 16.8 | 18.3 | 2.15 | 8.9 | 0.419 | 2.15 |
| ASCON | 34.8 | 37.5 | 3.31 | 7.7 | 2.664 | 3.31 |
| CLOC-AES | 33.1 | 36.4 | 2.67 | 10.0 | 5.327 | 2.67 |
| CLOC-TWINE | 71.6 | 86.2 | 6.95 | 20.1 | 7.848 | 6.95 |
| SILC-AES | 23.7 | 30.0 | 2.24 | 26.6 | 3.796 | 2.24 |
| SILC-PRESENT | 25.3 | 28.5 | 2.60 | 13.0 | 2.526 | 2.60 |
| SILC-LED | 40.2 | 44.5 | 3.70 | 10.6 | 6.162 | 3.70 |
| JAMBU-AES | 17.8 | 19.2 | 1.90 | 7.9 | 5.702 | 1.90 |
| JAMBU-SIMON | 96.5 | 111.2 | 4.90 | 15.2 | 2.614 | 4.90 |
| KETJE JR | 105.3 | 128.7 | 4.86 | 22.2 | 0.658 | 4.77 |

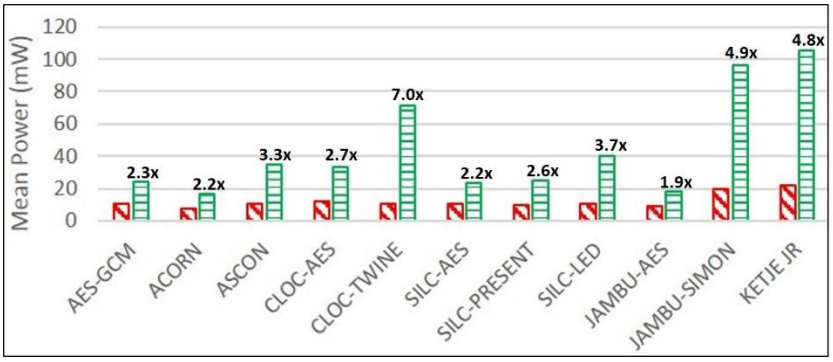


Fig. 21. Mean power of unprotected (using diagonal lines) and protected (using horizontal lines) cipher versions, measured as described in subsections 3.2 and 5.6. The relative increase is shown above protected versions.

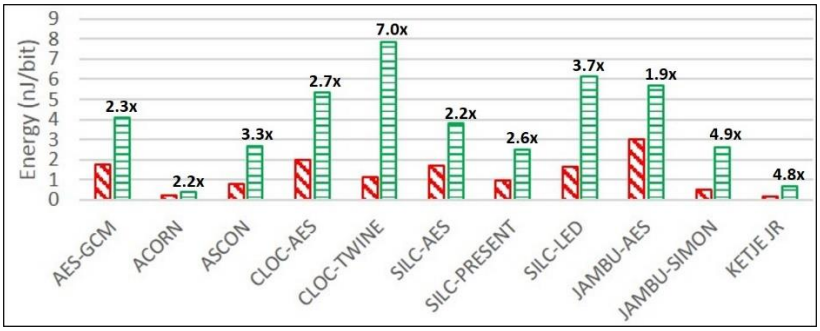


Fig. 22. Energy-per-bit of unprotected (using diagonal lines) and protected (using horizontal lines) cipher versions, measured as described in subsections 3.2 and 5.6. The relative increase is shown above protected versions.

# 6 Analysis

## 6.1 Absolute Costs of Protection against 1st Order DPA

For both the unprotected and protected implementations, ACORN is the smallest in terms of LUTs, followed by JAMBU-AES and JAMBU-SIMON. CLOC-AES and SILC-AES are larger than JAMBU-AES, since the CLOC and SILC implementations at [15] instantiate two AES cores, whereas JAMBU-AES uses only one. AES-GCM (with one AES core) is nearly the size of SILC-AES (with two AES cores), since the AES-GCM GF($2^{128}$) multiplier compares in size to the 8-bit pipelined AES core. Ketje Jr. and ASCON are relatively large due to their full-width basic-iterative architectures.

In terms of throughput, Ketje Jr. is highest among both unprotected and protected versions, followed by ACORN and JAMBU-SIMON. However, ACORN has the highest TP/A ratio, followed by Ketje Jr. and SIMON-JAMBU, for both unprotected and protected versions.

ACORN, followed by JAMBU-AES and SILC-PRESENT, have the lowest mean power consumption, as measured on the Spartan-6 FPGA at 10 MHz. For protected versions, ACORN uses the lowest mean power, followed by JAMBU-AES and SILC-AES.

Protected implementations resistant to DPA are generally not "constant-power" implementations. However, a minimal difference between $P_{mean}$ and $P_{max}$ is desirable, from both an engineering standpoint, and for reducing potential vulnerability to power analysis attacks. ASCON, JAMBU-AES, and ACORN have the lowest difference between $P_{mean}$ and $P_{max}$, while SILC-AES, Ketje Jr., and CLOC-TWINE have the greatest difference.

Ketje Jr. is the most energy-efficient of the unprotected cipher implementations, followed by ACORN and JAMBU-SIMON. For protected ciphers, ACORN is the most efficient, followed by Ketje Jr. and SILC-PRESENT.

## 6.2 Relative Costs of Protection against 1st Order DPA

The average number of LUTs increases by a factor of 3.1, and the throughput decreases by a factor of 1.8, when comparing unprotected to protected implementations. The reduction in throughput results from a 1.8 factor in average maximum frequency, which is due to increase in critical path and routing congestion in the protected versions. The average TP/A ratio of the protected implementations decreases by a factor of 5.6 compared to the unprotected versions. The average power and energy-per-bit of protected implementations increase by a factor of 3.4 compared to unprotected implementations.

However, the growth factor for area, and reduction factors for TP and TP/A ratios (respectively) for individual protected cipher versions vary widely. In terms of area (LUTs), protected versions of SILC-PRESENT, CLOC-AES, and SILC-LED have the lowest growth factors over unprotected versions, while ACORN, CLOC-TWINE, and Ketje Jr. have the highest growth factors. The reason for high area growth factors is a combination of architecture required for protection against DPA, and additional required randomness. For example, the high growth factor in ACORN is due to the addition of a PRNG capable of sourcing 120 random bits per clock cycle, the size of which is comparable to the area of the protected ACORN not including the PRNG.

In terms of throughput, the lowest reduction ratios for protected cipher versions are for JAMBU-AES, CLOC-AES, and AES-GCM, while the highest reduction ratios are for CLOC-TWINE, JAMBU-SIMON, and SILC-LED. Since architectures for protected and unprotected versions are analogous, this means that DPA protection most negatively affects the combination of critical path and routing congestion for CLOC-TWINE, JAMBU-SIMON and SILC-LED, and least affects JAMBU-AES, CLOC-AES, and AES-GCM. If we expand lowest reduction cost to fourth place, we note that SILC-AES and ACORN have nearly equivalent costs. This shows that the 8-bit pipelined AES core itself has a relatively low cost of protection against DPA.

In terms of throughput-to-area (TP/A) ratio, the lowest reduction ratios for protected ciphers are CLOC-AES, JAMBU-AES and AES-GCM, and the highest reduction ratios are CLOC-TWINE, ACORN, and JAMBU-SIMON. If we expand highest reduction ratios to four places, Ketje. Jr. has the next highest reduction cost. This shows that the best three overall performing protected ciphers (Ketje Jr., ACORN, and JAMBU-SIMON) also have the highest relative protection costs. CLOC-TWINE has the highest ratio, indicating that either our protection of the TWINE primitive, or implementation of the protected 3-share CLOC-TWINE, is sub-optimal and could be improved.

JAMBU-AES, ACORN, and SILC-AES have the lowest growth ratios in power and energy consumption comparing protected to unprotected versions, while CLOC-TWINE, JAMBU-SIMON, and Ketje Jr. have the highest growth ratios. This is a positive result for ACORN, since the highest performing protected cipher version (in terms of TP/A ratio) also has a very low growth in power consumption, at least at 10 MHz. While we have already noted the possibly sub-optimal DPA protection used in CLOC-TWINE, the high power and energy growths of JAMBU-SIMON and Ketje Jr. are explained by the use of architectures optimized for TP/A ratio (i.e., full-width datapath with basic iterative architectures), since the additional overhead of TI-protected modules results in the use of more than 5 times the additional computations in the same clock cycle compared to unprotected versions.

Table 5 ranks all authenticated ciphers in this study, in terms of absolute and relative costs of protections, as described above.

Table 5. Rankings of Ciphers in terms of Absolute and Relative Costs of Protection

| Cipher | LUT | Area Fctr. | TP | TP Fctr. | TP/A | TP/A Fctr. | Pwr | Pwr Fctr. | E/bit | E/bit Fctr. |
|---|---|---|---|---|---|---|---|---|---|---|
| Unprotected | | | | | | | | | | |
| AES-GCM | 6 | - | 8 | - | 8 | - | 4 | - | 9 | - |
| ACORN | 1 | - | 2 | - | 1 | - | 1 | - | 2 | - |
| ASCON | 9 | - | 4 | - | 4 | - | 6 | - | 4 | - |
| CLOC-AES | 11 | - | 10 | - | 11 | - | 9 | - | 10 | - |
| CLOC-TWINE | 5 | - | 6 | - | 6 | - | 5 | - | 6 | - |
| SILC-AES | 7 | - | 9 | - | 9 | - | 7 | - | 8 | - |
| SILC-PRESENT | 10 | - | 5 | - | 5 | - | 3 | - | 5 | - |
| SILC-LED | 8 | - | 7 | - | 7 | - | 8 | - | 7 | - |
| JAMBU-AES | 2 | - | 11 | - | 10 | - | 2 | - | 11 | - |
| JAMBU-SIMON | 3 | - | 3 | - | 3 | - | 10 | - | 3 | - |
| KETJE JR | 4 | - | 1 | - | 2 | - | 11 | - | 1 | - |
| Protected | | | | | | | | | | |
| AES-GCM | 7 | 5 | 6 | 3 | 6 | 3 | 4 | 4 | 7 | 4 |
| ACORN | 1 | 11 | 2 | 4 | 1 | 10 | 1 | 2 | 1 | 2 |
| ASCON | 10 | 8 | 4 | 7 | 5 | 7 | 7 | 7 | 5 | 7 |
| CLOC-AES | 9 | 2 | 7 | 2 | 10 | 1 | 6 | 6 | 8 | 6 |
| CLOC-TWINE | 11 | 10 | 8 | 11 | 11 | 11 | 9 | 11 | 11 | 11 |
| SILC-AES | 8 | 4 | 9 | 5 | 8 | 4 | 3 | 3 | 6 | 3 |
| SILC-PRESENT | 4 | 1 | 5 | 8 | 4 | 5 | 5 | 5 | 3 | 5 |
| SILC-LED | 5 | 3 | 10 | 9 | 9 | 6 | 8 | 8 | 10 | 8 |
| JAMBU-AES | 2 | 6 | 11 | 1 | 7 | 2 | 2 | 1 | 9 | 1 |
| JAMBU-SIMON | 3 | 7 | 3 | 10 | 3 | 9 | 10 | 10 | 4 | 10 |
| KETJE JR | 6 | 9 | 1 | 6 | 2 | 8 | 11 | 9 | 2 | 9 |

"Fctr" indicates growth ratio of Protected vs. Unprotected (Area, Power, or Energy-per-bit) or reduction ratio of Unprotected vs. Protected (Throughput or Throughput-to-area ratio). "TP" is throughput, "TP/A" is throughput-to-area ratio, "Pwr" is Power, and "Eng" is energy-per-bit. Rankings in terms of lowest area (LUT), lowest area growth ratio, highest TP, lowest TP reduction ratio, highest TP/A, lowest TP/A reduction ratio, lowest power, lowest power growth ratio, lowest energy-per-bit, and lowest energy-per-bit growth ratio.

### 6.3 Cipher Versions with Sub-optimal Cost of Protection

In this research, we examine implementations of CAESAR Round 3 candidate authenticated ciphers which are fully (or nearly-fully) compliant with the CAESAR HW API for Authenticated Ciphers [5, 6]. The version of ACORN at [16] enables a close-to-optimal protection against DPA using threshold implementations, since it uses a small datapath width (i.e., 8 bits), and has a maximum of two cascaded `and` gates in its non-linear state update computation path. In general, however, the implementations available at [15] and [17 – 19] are not optimized for TI protection. Specifically, they have either large datapath widths (e.g., 128 bits for AES-based ciphers, 64 bits for ASCON, SILC-PRESENT, SILC-LED, CLOC-TWINE, etc.), basic iterative architectures with multiple non-linear operations performed in parallel (e.g., ASCON, CLOC-SILC cipher variants, Ketje Jr.), or even unrolled architectures with multiple rounds completed in a single cycle (e.g., JAMBU-SIMON).

While the above choices of architecture provide optimal throughput-to-area (TP/A) ratios, they are suboptimal when attempting TI-protection. Some reasons include:

1. Wide datapaths with multiple TI-protected gates in the same clock cycle lead to large growth of resources (which increase quadratically in order of protection), and large power consumption, which is not optimal for IoT devices.

2. Multiple cascaded non-linear computations, occurring in the same clock cycle, increase the probability of enabling power correlations based on glitch transitions in CMOS logic, which have the potential to leak sensitive information [30].

3. The amount of randomness (measured in random bits per clock cycle) required for resharing from two to three TI shares, or required to meet the TI uniformity property, increases with wide datapaths and with basic iterative or unrolled architectures. This increases the required output of either an internal randomness source (such as a PRNG), or external randomness provided through an interface.

Therefore, authenticated ciphers, optimized for TI protection, should be constructed with small internal datapaths (e.g., 8 or 16 bits), and with a maximum of one logic level of non-linear functions (e.g., `and`) conducted in a single clock cycle, which could result in pipelined or folded (e.g., multi-cycle) architectures. This approach has been fully adopted for modification of AES-based ciphers (i.e., reduced datapath and pipelined architecture), and partially adopted for ACORN and ASCON (e.g., multi-cycle architectures). However, these techniques should be investigated for all authenticated cipher candidates, and is left to future research.

## 7 Conclusions

In this research we introduced a methodology for conducting t-test leakage detection on authenticated ciphers, in order to determine resistance to DPA side-channel attack, and to verify effectiveness of countermeasures against DPA. Our methodology, which leverages the open-source FOBOS test bench, CAESAR Hardware API standards, and related Development Package, shows that unprotected implementations of AES-GCM, ACORN, ASCON, CLOC (AES and TWINE), SILC (AES, PRESENT, and LED), JAMBU (AES and SIMON), and Ketje Jr., in the Spartan-6 FPGA, are likely vulnerable to DPA.

We implement protected versions of all 11 ciphers, and verify their improved resistance to 1st order DPA using the t-test methodology. Additionally, using the t-test, we demonstrate a limitation in the t-test leakage detection methodology for authenticated ciphers, due to a data-dependent conditional decision in the CLOC specification. Although CLOC-AES and CLOC-TWINE protected implementations do not pass a t-test with generic test vectors, we use modified test vectors to demonstrate that the protected CLOC authenticated ciphers achieve improved resistance to 1st order DPA.

ACORN has the lowest area (in terms of LUTs) of protected ciphers, followed by JAMBU-AES and JAMBU-SIMON. Likewise, ACORN has the highest throughput-to-area (TP/A) ratio, followed by Ketje Jr. and JAMBU-SIMON. ACORN is also the most energy efficient of the protected implementations (i.e., uses the lowest energy-per-bit), followed by Ketje Jr. and SILC-PRESENT, according to our evaluation on the Spartan-6 FPGA at a fixed frequency of 10 MHz.

In terms of costs of protection against 1st order DPA, the area (in LUTs) increases by an average factor of 3.1, the throughput decreases by a factor of 1.8, and the TP/A ratio decreases by a factor of 5.6, when comparing protected to unprotected implementations. The energy-per-bit of protected implementations increases by an average factor of 3.4 compared to unprotected implementations.

SILC-PRESENT has the lowest relative growth in area, while JAMBU-AES has the lowest reduction in throughput, and CLOC-AES has the lowest reduction in TP/A ratio, when comparing protected to unprotected cipher versions. JAMBU-AES has the lowest growth in power and energy-per-bit.

## 8 Areas for Future Research

Future research could include investigation of additional pairs of authenticated ciphers, investigation of cipher versions which are optimized for protection against DPA, and measurement of power and energy at higher frequencies, i.e., closer to actual maximum operating frequencies. The use of attack-based testing methods (such as Correlation Power Analysis) to quantify improved resistance of protected versions to DPA (including higher orders of DPA) could provide additional insight into the relative costs of protection of the subject ciphers. Additionally, the techniques in this research could be adapted to investigate costs of protection for future cryptographic competitions, such as for post-quantum resistant public key cryptography.

## Acknowledgement

This research is based on work supported by the National Science Foundation under Grant No. 1314540.

## References

1. "CAESAR: Competition for Authenticated Encryption: Security, Applicability, and Robustness." Internet: http://competitions.cr.yp.to/caesar.html.
2. D. Bernstein, 2016, Jul 17, "Cryptographic Competitions," Google Groups, https://groups.google.com/forum/#!forum/crypto-competitions.
3. G. Goodwill, B. Jun, J. Jaffe and P. Rohatgi, "A testing methodology for side channel resistance validation," in NIST Non-invasive Attack Testing Workshop, 2011.
4. R. Velegalati and J.P. Kaps, "Towards a Flexible Opensource Board for Side-channel analysis (FOBOS)," in Cryptographic architectures embedded in reconfigurable devices, CRYPTARCHI 2013, Jun, 2013.
5. E. Homsirikamol, W. Diehl, A. Ferozpuri, F. Farahmand, P. Yalla, J.P. Kaps and K. Gaj, "CAESAR Hardware API," Cryptology ePrint Archive, Report 2016/626, Internet: http://eprint.iacr.org/2016/626.pdf Jun. 14, 2016.
6. E. Homsirikamol, W. Diehl, A. Ferozpuri, F. Farahmand, P. Yalla, J.P. Kaps and K. Gaj, "Addendum to the CAESAR Hardware API v1.0," Internet: https://cryptography.gmu.edu/athena/CAESAR_HW_API/CAESAR_HW_API_v1.0_Addendum.pdf, Jun. 17, 2016.
7. George Mason University, "Development Package for the CAESAR Hardware API, v2.0," https://cryptography.gmu.edu/athena/index.php?id=CAESAR, Dec. 5, 2017.
8. W. Diehl and K. Gaj, "RTL Implementations and FPGA Benchmarking of Selected CAESAR Round Two Authenticated Ciphers," Microprocessors and Microsystems, vol. 52, Jul. 2017, pp. 202-218.
9. H. Wu, "ACORN, a lightweight authenticated cipher (v3)," Sep, 15, 2016, Internet: https://competitions.cr.yp.to/round3/acornv3.pdf

10. C. Dobraunig, M. Eichlseder, F. Mendel, M Schläffer "ASCON v1.2," Sep. 15, 2016, Internet: https://competitions.cr.yp.to/round3/asconv12.pdf
11. T. Iwata, K. Minematsu, J. Guo, S. Morioka, and E. Kobayashi, "CLOC and SILC v3," https://competitions.cr.yp.to/round3/clocsilcv3.pdf, Sep. 2016
12. H. Wu and T. Huang, "The JAMBU Lightweight Authenticated Encryption Mode," Sep. 15, 2016, Internet: http://www3.ntu.edu.sg/home/wuhj/research/caesar/caesar.html
13. G. Bertoni, J. Daemen, M. Peeters, G. Van Assche and R. Van Keer, "CAESAR Submission: Ketje V2," https://competitions.cr.yp.tp/round3/ketjev2.pdf," Sep. 2016
14. D. McGrew and J. Viega, "The Galois/Counter Mode of Operation (GCM)," *Submission to NIST Modes of Operation Process*, Jan. 04.
15. CERG GMU Source Code of CAESAR Round 3 Candidates, 2017, Aug. 8, https://cryptography.gmu.edu/athena/index.php?id=CAESAR_source_codes.
16. T. Huang, "Round 3 hardware submission: ACORN," Internet: https://groups.google.com/forum/#!topic/crypto-competitions, Jul. 2017.
17. T. Iwata, "HW for CLOC and SILC 64-bit BC," Internet: https://groups.google.com/forum/#!topic/crypto-competitions, Jul. 2017.
18. T. Huang, "SIMON-JAMBU" Internet: https://groups.google.com/forum/#!forum/crypto-competitions, Aug. 2017.
19. G. Bertoni, "Ketje-Keyak Team," Internet: https://github.com/guidobertoni/caesar_gmu_vhdl [Dec. 6, 2017]
20. P. C. Kocher, J. Jaffe, and B. Jun, "Differential Power Analysis," Proceedings of CRYPTO '99 - 19th International Conference on Cryptology, Aug. 15-19, 1999, Santa Barbara, CA.
21. P. Kocher, J. Jaffe, B. Jun, and P. Rohatgi, "Introduction to differential power analysis," C. Kaya Koc, ed., Journal of Cryptographic Engineering, vol. 1, no. 1, pp. 5-27, Apr. 2011.
22. W. Diehl, A. Abdulgadir, J. P. Kaps and K. Gaj," Comparing the Cost of Protecting Selected Lightweight Block Ciphers Against Differential Power Analysis in Low-Cost FPGAs," International Conference on Field Programmable Technologies (ICFPT 2017), Dec. 11-15, 2017, Melbourne, Australia, pp. 128-135.
23. T. Schneider and A. Moradi, "Leakage assessment methodology", Journal of Cryptographic Engineering, vol. 6, no. 2, pp. 85-89, Jun. 2016.
24. A. Shahverdi, M. Taha and T. Eisenbarth, "Lightweight Side Channel Resistance: Threshold Implementations of Simon," in IEEE Transactions on Computers, vol. 66, no. 4, pp. 661-671, Apr. 1 2017.
25. F. Bache, T. Schneider, A. Moradi and T. Güneysu, "SPARX — A side-channel protected processor for ARX-based cryptography," in Design, Automation & Test in Europe Conference & Exhibition (DATE), 2017, Lausanne, 2017, pp. 990-995.
26. H. Groß and Stefan Mangard, "Reconciling d + 1 Masking in Hardware and Software," https://eprint.iacr.org/2017/103, Jun. 2017.
27. S. Nikova, C. Rechberger, V. Rijmen, "Threshold Implementations Against Side-Channel Attacks and Glitches," Information and Communications Security, Volume 4307 of the series Lecture Notes in Computer Science pp. 529-545, 2006
28. A. Shamir. "How to Share a Secret." Communications of the ACM, Vol. 22 No. 11, 1979, pp. 612-613.
29. A. Yao, "Protocols for Secure Computation," FOCS 1982, pp. 160-164
30. S. Mangard, N. Pramstaller, E. Oswald, "Successfully attacking masked AES hardware implementations" In: Rao, J.R., Sunar, B. (eds.) CHES 2005. LNCS, vol. 3659, pp. 157–171. Springer, Heidelberg (2005)
31. B. Bilgin, B. Gierlichs, S. Nikova, V. Nikov, V. Rijmen, "A More Efficient AES Threshold Implementation," eds. D. Pointcheval, D. Vergnaud, Progress in Cryptology -- AFRICACRYPT 2014: 7th International Conference on Cryptology in Africa, Marrakesh, Morocco, May 28-30, 2014. Proceedings, 2014, Springer International Publishing, pp. 267—284
32. A. Moradi, A. Poschmann, S. Ling, C. Paar, H. Wang, "Pushing the Limits: A Very Compact and a Threshold Implementation of AES," ed. K. Paterson, Advances in Cryptology -- EUROCRYPT 2011: 30th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Tallinn, Estonia, May 15-19, 2011. Proceedings, 2011, Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 69—88
33. E. Homsirikamol, W. Diehl, A. Ferozpuri, F. Farahmand, and K. Gaj "Implementer's Guide to the CAESAR Hardware API v2.0" https://cryptography.gmu.edu/athena/CAESAR_HW_API/CAESAR_HW_Implementers_Guide_v2.0.pdf Dec. 2017.
34. CERG GMU, "Flexible Open-source workBench fOr Side-channel analysis (FOBOS)," Internet: https://cryptography.gmu.edu/fobos/ 2016, Oct. 25.
35. R. Kern, "A Simple File Format for NumPy Arrays," Dec. 20, 2007, https://docs.scipy.org/doc/numpy-dev/neps/npy-format.html
36. J. Vliegen, O. Reparaz and N. Mentens, "Maximizing the throughput of threshold-protected AES-GCM implementations on FPGA," *2017 IEEE 2nd International Verification and Security Workshop (IVSW)*, Thessaloniki, 2017, pp. 140-145.
37. D. Canright, L. Batina, "A Very Compact 'Perfectly Masked' S-Box for AES," Applied Cryptography and Network Security, Volume 5037 of the series Lecture Notes in Computer Science pp. 446-459, 6th International Conference, ACNS 2008, New York, NY, USA, June 3-6, 2008.

38. K. Gaj, P. Chodowiec, “FPGA and ASIC Implementations of AES,” In Ç. K. Koç (ed.) Cryptographic Engineering, Springer Science & Business Media, 2009, pp. 235-294.
39. N. Ferguson, “Authentication Weaknesses in AES-GCM,” Microsoft Corporation, May. 5, 2005, Internet: https://csrc.nist.gov/csrc/media/projects/block-cipher-techniques/documents/bcm/comments/cwc-gcm/ferguson2.pdf
40. J. Jaffe, "A First-Order DPA Attack Against AES in Counter Mode with Unknown Initial Counter,"Cryptographic Hardware and Embedded Systems - CHES 2007: 9th International Workshop, Vienna, Austria, Sep. 10-13, 2007, pp. 1-13
41. S. Belaid, P. Fouque, B. Gerard, "Side Channel Analysis of Multiplications in GF($2^{128}$)," ASIACRYPT 2014, Part II, LNCS vol. 8874, 2014, pp. 306-325.
42. A. Poschmann, A. Moradi, K. Khoo, C. Lim, H. Wang and S. Ling, “Side-Channel Resistant Crypto for Less than 2,300 GE,” Journal of Cryptology, 2011, vol. 24, pp. 322-345.
43. S. Kutzner, P. Nguyen, A. Poschmann and H. Wang, “On 3-Share Threshold Implementations for 4-Bit S-boxes,” Constructive Side-Channel Analysis and Secure Design: 4th International Workshop, COSADE 2013, Paris, France, Mar. 6-8, 2013, Revised Selected Papers, 2013, pp. 99-113.
44. M. Rivain and E. Prouff, “Provably Secure Higher-Order Masking of AES,” 12th International Workshop on Cryptographic Hardware and Embedded Systems, CHES 2010, Santa Barbara, USA, Aug. 17-20, 2010, LNCS, vol. 6225, pp 413-427.
45. CERG GMU, “Automated Tool for Hardware Evaluation (ATHENa)”, Internet: https://cryptography.gmu.edu/athena/ [Mar. 18, 2017]. “CAESAR: Competition for Authenticated Encryption: Security, Applicability, and Robustness,” http://competitions.cr.yp.to/ caesar.html.