Zero-Sum Partitions of PHOTON Permutations

Qingju Wang[1], Lorenzo Grassi[2], Christian Rechberger[1,2]

[1] Technical University of Denmark, Denmark,
[2] IAIK, Graz University of Technology, Austria
`quwg@dtu.dk, lorenzo.grassi@tugraz.at, christian.rechberger@tugraz.at`

Abstract. We describe an approach to zero-sum partitions using Todo's division property at EUROCRYPT 2015. It follows the inside-out methodology, and includes MILP-assisted search for the forward and backward trails, and subspace approach to connect those two trails that is less restrictive than commonly done.
As an application we choose PHOTON, a family of sponge-like hash function proposals that was recently standardized by ISO. With respect to the security claims made by the designers, we for the first time show zero-sum partitions for almost all of those full 12-round permutation variants that use a 4-bit S-Box. As with essentially any other zero-sum property in the literature, also here the gap between a generic attack and the shortcut is small.
Key words: PHOTON, Integral, Division Property, Zero-sum, MILP, Subspace

1 Introduction

Hash functions are one of the most important primitives in symmetric-key cryptography. Sponge functions [6] are a way of building hash functions from a fixed permutation. Modern cryptanalytic approaches target both hash function primitives and underlying ciphers or permutations. Internal components are indeed expected to provide certain properties and for verifying their closeness to ideal behavior it is important to evaluate the security of hash functions. The analysis of hash functions underlying block ciphers or permutations is often done in the known-key model, as introduced by Knudsen and Rijmen in [21].

In this paper, we exhibit the very first zero-sum partitions based on the integral property on the full permutation of some PHOTON variants, a lightweight hash function proposed by Guo *et al.* [19] at CRYPTO 2011 and recently standardized by ISO.

1.1 Background on the Integral Distinguishers, Zero-sum Distinguishers and Division Property

Integral and Zero-Sum Distinguishers. A possible analysis of the inner permutation of a hash function is based on the *zero-sum property*, which can be seen as a generalization of an integral property [22]. The integral attack, also known as square attack, is originally proposed by Knudsen to analyze SQUARE block cipher [11]. There are several variants of the integral attack with different names: multiset attack [7], saturation attack [24], and collision attack [16]. Integral distinguishers mainly make use of the observation that when fixing some parts of the plaintext, the specific parts of the ciphertext have balanced property, i.e. each possible partial value occurs the exact same number of times in the output.

In more details, a zero-sum structure for a function f is defined as a set Z of inputs z_i that sum to zero, and for which the corresponding outputs $f(z_i)$ also sum to zero (see Aumasson and Meier[3]). For an iterated function, the existence of many zero-sums is usually due either to the particular structure of the round function or to a low degree. Since it is expected that a randomly chosen function does not have many zero-sums, the existence of several such sets of inputs can be seen as a distinguishing property of the internal function. By using the *inside-out* technique, zero-sums could be constructed starting from the middle, and be extended to forward and backward direction as far as possible.

Division Property. As we have already said, a zero-sum property can be found working on the degree of the function. As an example, if f is a k-degree function on $\mathbb{F}_{2^n}$, then it is proved that $\bigoplus_{v \in V \oplus a} f(v) = 0$ for any $(k+1)$-dimension subspace $V \subseteq \mathbb{F}_{2^n}$ where $V \oplus a$ is an arbitrary coset of V (see *Higher Order Differential* [23] for details). The main approach to construct zero-sum distinguishers is related to find accurate estimations on the degree of both the forward and backward permutations that define the encryption/hash function f.

As a generalized integral property, *division property* was proposed by Todo at EUROCRYPT 2015 [28] to search integral distinguishers for symmetric-key primitives including SPNs and Feistel structures. Taking SPNs - which are also the main focus of this paper - as an example, the main idea was to formulate the propagation of division property through an S-Box, where the S-Box was regarded as unknown but restricted only by its algebraic degree. Moreover, since the degree remains the same while going through linear permutations, division property propagation through the permutation layer can easily be modeled. Based on these, new integral distinguishers for many SPN ciphers have been constructed. One prominent example was the application to MISTY1 [27], where the S-Box S_7 was shown to have an important vulnerability in terms of division property. By employing this, a new 6-round integral distinguisher was constructed, and a full-round attack on MISTY1 was achieved for the first time. At CRYPTO 2016, Boura and Canteaut [9] proposed a new notion, called *parity set*, to study division property from the coding theory('s) point of view, based on which they found better integral distinguishers for PRESENT.

Motivated by narrowing the 5 rounds gap between the integral distinguishers for SIMON32 in [31] and [28], bit-based division property [29] was introduced at FSE 2016, where the division property of each bit was treated independently. As a result, the 14-round integral distinguishers for SIMON32 in [31] were found. However, as pointed out in [29], for a block cipher with block size n, the time and memory complexity is lower bounded by 2^n. As most ciphers adopt block size larger than 32, this makes searching integral distinguisher by bit-based division property under this framework computationally infeasible.

To solve this problem, Xiang *et al.* [32] built an automatic tool based on mixed integer linear programming (MILP) to study the division property of SPNs with bit-permutation linear layers (e.g. PRESENT). They first introduced notion *division trail* to build the objective function, then represented the operations of the ciphers by linear (in)equalities to constrain the objective function. After setting the required stopping rules of searching division trails, they could determine the existence of certain number of rounds integral distinguishers by optimizing the MILP. As a result, they found many interesting integral distinguishers for the targeted ciphers. Later, a MILP automatic tool for SPNs with non-bit-permutation linear layers (mainly MDS matrices) was studied in [25].

It shows that an automatic tool based on bit-based division property is very powerful in the search of better integral distinguishers, and therefore we for the first time apply it to construct zero-sum distinguishers for symmetric-key primitives which we will pursue in the following.

1.2 Our Contributions

In this paper we focus on zero-sum distinguishers exploiting the recent division property developments in the searching integral distinguishers and provide much improved results compared to earlier works. As an application, we choose the internal permutation of PHOTON, which is a lightweight hash function proposed by Guo *et al.* [19] at CRYPTO 2011 and has been standardized in ISO/IEC 29192-5:2016, to demonstrate our new techniques. As the "idealness" of the underlying permutation is important for security properties that are expected for a sponge-based hash function using it, the PHOTON designers claim particular security levels for each of their variants. We exhibit for the first time distinguishers of the full number of rounds with a complexity that is below the designers claims[1] (except the one with internal state size 100, for which we present a distinguisher on 11 out of 12 rounds).

Our concrete results are summarized in Table 1. For achieving these results, we use an approach that combines various ideas from related areas which we summarize in the following:

Inside-out approach for division-property distinguishers. The inside-out approach was perhaps first used by Wagner [30] in block cipher cryptanalysis or Dobbertin [12,13] with his work on the MD5 compression function. It later became a default approach for analyzing various building blocks in symmetric cryptography. We for the first time apply this approach to distinguishers using the division property.

MILP automatic tool to search zero-sum partitions based on division property. We mainly focus on versions with 4-bit S-Box (PRESENT S-Box).

[1] We mention that our distinguishers have only a small advantage (approximately a factor 2) when compared to the generic attack.

We find a set of 8 linear equations to represent its division trails table, which is 3 less than the one in [32]. This enables us to obtain 6-round zero-sum partitions for versions of PHOTON permutation with 4-bit S-Box, which are not given in [25]. A detailed description of the automatic tool can be found in Sect. 3.1.

Improved zero-sum partitions based on the weakness of PRESENT S-Box. We further exploit the algebraic normal form (ANF) of the PRESENT S-Box, and find that when fixing the least significant bit of the input, the algebraic degree drops from 3 to 2. This property of the S-Box enable us to find zero-sum partitions based on division property that can improve some of the results in [28] by 2 of data complexity. Since we look at integral property of both forward and backward directions of the internal permutations, though this advantage of our distinguishers for single forward/backward direction is not substantial, when constructing a zero-sum distinguisher, we can directly gain an advantage of 2^2 in size of partitions. The detailed zero-sum partitions of PHOTON permutations are provided in Sect. 5 and App. C–E.

A method to add one round in the middle. Using the MILP automatic tool just cited, an attacker can find initial set of texts with active/partial active/constant nibbles that satisfy the zero-sum property after a certain number of decryption - encryption rounds. For the decryption case and for some of these sets, we choose how to keep/preserve (almost) for free this property adding one round at the beginning. The basic idea is to choose sets for which (1) some linear relations (which depend on the MixColumns matrix) hold between the nibbles that lie on the same column - a property/case which is not investigated by the MILP automatic tool - and (2) that are mapped in the sets found by the automatic tool one round before. Such a strategy can be easily described using the subspace trail notation [18]. Those sets are finally used by the inside-out approach in order to set up the zero-sum partition. All details are given in Sect. 5.3.

When using the subspace trail to connect two initial zero-sum partitions for both directions, we also present a generic formula, without writing out the representation of the state, to compute directly the dimension of combined middle round from the dimensions of the two initial subspaces.

We note that such strategy to add one round in the middle is not new in literature. A similar technique is exploited for example by Gilbert in [15] in order to set up a 8-round integral known-key distinguisher extending the 7-round initial proposed by Knudsen and Rijmen [21]. However, while Gilbert explains such result using the super-S-Box notation, we present it using the subspace trail cryptanalysis. It turns out to be directly applicable for the distinguishers found by the MILP tool. Finally, other (different) techniques to gain rounds in the middle have been proposed e.g. by Boura and Canteaut in [8] to set up a 18-round distinguisher for Keccak.

Zero-sums for hash function PHOTON We apply our zero-sum approach to the hash functions. Because the utilization of degrees of freedom for PHO-

Table 1. *PHOTON-$n/r/r'$ Permutation Distinguishers.* We list here the currently best known results on inner permutations. All variants have full 12 rounds and we focus on the variants with 4-bits S-Box. "Partition Size N" denotes the size of the zero-sum partitions.

PHOTON Variants	**Security Claim**	**# Rounds**	**Partition Size N**	**Property**	**Reference**
-80/20/16	80	8	2^{8}	Multiple Diff. Trail	[19]
		9	2^{35}	Partial Balance	App. C.1
		9	2^{36}	Balance	App. C.1
		10	2^{40}	Balance	App. C.1
		11	2^{76}	Balance	App. C.2
-128/16/16	128	8	2^{8}	Multiple Diff. Trail	[19]
		9	2^{42}	Balance	Sect. 3.3
		10	2^{47}	Balance	Sect. 5.1
		11	2^{107}	Balance	Sect. 5.2
		12	$\mathbf{2^{127}}$	**Partial Balance**	**Sect. 5.3**
-160/36/36	160	8	2^{8}	Multiple Diff. Trail	[19]
		9	2^{43}	Partial Balance	App. D.1
		9	2^{44}	Balance	App. D.1
		10	2^{48}	Balance	App. D.1
		11	2^{108}	Balance	App. D.2
		12	$\mathbf{2^{159}}$	**Partial Balance**	**App. D.3**
-224/32/32	224	8	2^{8}	Multiple Diff. Trail	[19]
		9	2^{184}	Parallel Merging	[20]
		9	2^{50}	Balance	App. E.1
		10	2^{54}	Balance	App. E.1
		11	2^{119}	Balance	App. E.2
		12	$\mathbf{2^{184}}$	**Balance**	**Sect. 5.4**

TON is so thin that we can only create 4-round zero-sums for almost all of the variants. We demonstrate our result on one example in Sect. 6.

2 A Brief Description of PHOTON

The domain extension algorithm of PHOTON is largely inspired from the sponge functions introduced by Bertoni *et al.* [6] in 2007. It uses sponge functions framework in order to keep the internal memory size as low as possible.

There are 5 variants of PHOTON and are denoted by PHOTON-$n/r/r'$, where n is the bit-size of the hash output, r and r' are input and output bitrate respectively. c is defined as the bit-size of the capacity part of the internal state, and $t = (c + r)$ is the internal state size. As a consequence, the 5 internal permutations are defined as P_t, where $t \in \{100, 144, 196, 256, 288\}$. The internal

Table 2. Parameters of PHOTON-$n/r/r'$

Versions	Permutation P_t	t	n	c	r	r'	d	s
PHOTON-80/20/16	P_{100}	100	80	80	20	16	5	4
PHOTON-128/16/16	P_{144}	144	128	128	16	16	6	4
PHOTON-160/36/36	P_{196}	196	160	160	36	36	7	4
PHOTON-224/32/32	P_{256}	256	224	224	32	32	8	4
PHOTON-256/32/32	P_{288}	288	256	256	32	32	6	8

state of the permutation is, similarly to the AES, viewed as a $(d \times d)$ matrix of s-bit cells and the corresponding values depending on t are given in Table 2.

In this paper we focus on the integral property of the internal permutation P_t, we therefore describe them in details. For the domain extension function we refer to [19]. Similar to the AES, four operations are applied to each round below

- AddConstants: applies round-dependent constants to each cell of the first column.
- SubCells: applies the s-bit S-Box to every cell of the internal state. PRESENT S-Box is chosen for P_t for $t \in \{100, 144, 196, 256\}$ while AES S-Box is preferred for P_{288}.
- ShiftRows: rotates each cell located at row i by i positions to the left.
- MixColumnsSerial: updates linearly all columns independently. The matrix underlying the MixColumnsSerial layer is Maximum Distance Separable (MDS) so as to provide maximal diffusion. We refer to [19] for matrix for each state size t.

Every PHOTON internal permutation iterates 12 rounds.

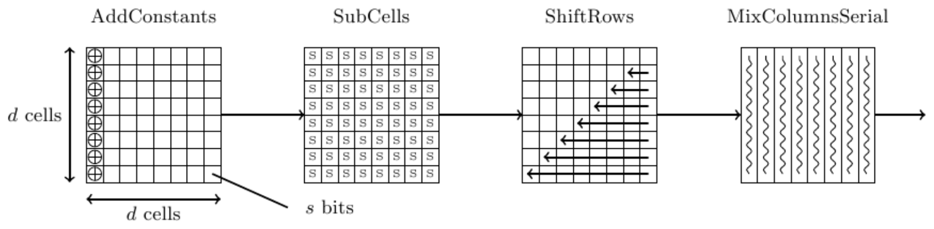

Fig. 1. One Round of PHOTON Internal Permutation

3 Zero-sum Partitions and Its Construction Based on Division Property by MILP

Definition 1. *(Zero-sum)[8] Let F be a function from $\mathbb{F}_{2^n}$ into $\mathbb{F}_{2^m}$. A zero-sum for F of size K is a subset $\{x_1, \ldots, x_K\} \subset \mathbb{F}_{2^n}$ of elements which sum to zero and for which the corresponding images by F also sum to zero, i.e., $\sum_{i=1}^{K} x_i = \sum_{i=1}^{K} F(x_i) = 0$.*

In general, given F a permutation over $\mathbb{F}_{2^n}$, a much stronger property - named *zero-sum partition* - can be investigated.

Definition 2. *(Zero-sum Partition)[8] Let P be a permutation from $\mathbb{F}_{2^n}$ to $\mathbb{F}_{2^n}$. A zero-sum partition for P of size $K = 2^k$ is a collection of 2^k disjoint $\{X_1, X_2, ..., X_k\}$ sets with the following properties:*

- *$X_i = \{x_{i,1}, \dots, x_{i,2^{n-k}}\} \subset \mathbb{F}_{2^n}$ for each $i = 1, ..., k$ and $\cup_{i=1}^{2^{n-k}} X_i = \mathbb{F}_{2^n}$;*
- *for each $i = 1, ..., 2^k$: $\sum_{x_{i,j} \in X_i} x_{i,j} = \sum_{x_{i,j} \in X_i} P(x_{i,j}) = 0$.*

We focus on creating zero-sum partitions of the permutation P. Assume P is an iterated permutation of the form $P = R_r \circ \cdots \circ R_1$, where all R_i is permutations over $\mathbb{F}_{2^n}$, named the round function of permutation P. Remember that for the permutation in a hash function, one can exploit any state starting from an intermediate state, without knowing any secret element. Assume one can find a set of texts $X = \{x^i\}_i$ and a set of texts $Y = \{y^i\}_i$ with the following properties:

$$\bigoplus_i R_r^{-1} \circ \cdots \circ R_{s+1}^{-1}(y^i) = 0 \qquad \text{and} \qquad \bigoplus_i R_1 \circ \cdots \circ R_s(x^i) = 0.$$

For the following, note that $X \oplus Y = \bigcup_{y \in Y} X \oplus y = \bigcup_{x \in X} Y \oplus x$, and $|X \oplus Y| = K$. Since one can work with the intermediate states, and simply chooses texts in $X \oplus Y$ and simply defines the plaintexts p_i as the $(r - s)$ rounds decryption of $X \oplus Y$, and the corresponding ciphertexts c_i as the s rounds encryptions of $X \oplus Y$. A zero-sum partition $\{p_i\}_{i=1,...,K}$ with the properties $\sum_{i=1}^{K} p_i = \sum_{i=1}^{K} c_i = 0$ is created for permutation P. We will follow this strategy to construct zero-sum partitions of PHOTON permutations in the following.

Notation - Zero-sum. First we introduce the notations that we are going to use to present our zero-sums. Let Λ be a collection of state vectors $X = (x_0, \dots, x_{2^n-1})$ where $x_i \in \mathbb{F}_{2^m}$.

- A: if all x_i in Λ are distinct, X is called active
- B: if the sum of all x_i in Λ can be predicted, X is called balanced
- C: if the values of x_i in Λ are equal, X is called passive/constant
- ?: if the sum of all x_i in Λ cannot be predicted, X is called unknown

When considering bit-level - i.e. let $x_i \in \mathbb{F}_2$ (the above m is equal to 1), we use lower case letters instead of uppercase letters, that is a represents an active bit, b a balance one, c a constant one and ? an unknown bit. For example, "*aaac*" in the nibble means that only the least significant bit is constant, all the others are active. Similarly, "???*b*" means that only the least significant bit is balanced, while the rest are unknown. For simplicity, we call a nibble with property "*aaac*" as partial active nibble, and "???*b*" as partial balance nibble in this paper.

Finally, we denote by $\mathbb{B}$ a full-balance state of size $d \times d$, and $\mathbb{PB}$ a partial-balance state of size $d \times d$.

3.1 Model Bit-based Division Property Propagation of Operations by MILP

In this section, we recall how to model the bit-based division property propagation of operations in a cipher by using MILP: copy, XOR, S-Box and Mix-Columns. Then we describe the searching strategy for zero-sum partitions based on MILP of division propagation. Some preliminaries of division property are provided in App. B.1, while we refer to [28] and [10] for a formal description of the division property.

Model Operations. We recall how to model the operations in ciphers to construct the MILP [32] - [33].

Model Copy. Let $\mathbb{X}$ be an input multiset of copy operation whose elements x take a value of $\mathbb{F}_2$, and $\mathbb{Y}$ be the output multiset whose elements (y_0, y_1) take a value of $\mathbb{F}_2 \times \mathbb{F}_2$. The copy operation creates $y = (y_0, y_1)$ from $x \in \mathbb{X}$ as $y_0 = x$ and $y_1 = x$. Assume the input multiset has division property $\mathcal{D}^1_k$ (since this is on bit-level, we do not distinguish between $\mathcal{D}^{1,1}_k$ and $\mathcal{D}^1_k$), then the corresponding output multiset has division property $\mathcal{D}^1_{(0,k),(1,k),\ldots,(k,0)}$. Since we consider bit-based division property, the input multiset division property $\mathcal{D}^1_k$ must have $0 \leq k \leq 1$. If $k = 0$, the output multiset has division property $\mathcal{D}^1_{(0,0)}$; otherwise, the output multiset has division property $\mathcal{D}^1_{(0,1)(1,0)}$. Thus, $(0) \xrightarrow{\text{copy}} (0,0)$ is the only division trail given the initial division property $\mathcal{D}^1_0$, and $(1) \xrightarrow{\text{copy}} (0,1)$, $(1) \xrightarrow{\text{copy}} (1,0)$ are the two division trails given the initial division property $\mathcal{D}^1_1$.

Let $a \xrightarrow{\text{copy}} (b_0, b_1)$ denote the division trail of the copy operation $x \xrightarrow{\text{copy}} (y_0, y_1)$, then MILP can describe this by the following inequality: $a - b_0 - b_1 = 0$, where $a, b_0, b_1 \in \{0,1\}$ are binaries.

Model XOR. Let $\mathbb{X}$ denote the input multiset whose elements $x = (x_0, x_1)$ take a value of $\mathbb{F}_2 \times \mathbb{F}_2$, and $\mathbb{Y}$ denote the output of the XOR operation where $y = x_0 \oplus x_1$. Assume the input multiset $\mathbb{X}$ has division property $\mathcal{D}^2_{\mathbf{k}}$ where $\mathbf{k} = (k_0, k_1)$, thus the corresponding output multiset $\mathbb{Y}$ has division property $\mathcal{D}^1_{k_0+k_1}$.

Let $(a_0, a_1) \xrightarrow{\text{XOR}} b$ denote a division trail through XOR operation $y = x_0 \oplus x_1$, which can be described by MILP through the following equality: $a_0 + a_1 - b = 0$ where $a_0, a_1, b \in \{0,1\}$.

Model S-Box. Xiang *et al.* [32] exploited the algebraic normal form (ANF) of an S-Box, and provided an accurate description the division trail (B.1) of an S-Box. For any S-Box, one can easily build the Division Trail Table (DTT) according to the definition of division property of boolean functions. Next we describe briefly how to model the DTT of an S-Box by MILP.

At ASIACRYPT 2014, Sun *et al.* [26] encoded the differential distribution table of an S-Box to the set of linear equations describing ciphers. The idea is to choose a set of linear (in)equalities $\mathcal{L}$ from the H-Representation of the convex hull of a set of points $\mathcal{A}$ in differential distribution table of an S-Box, such that the feasible solutions of $\mathcal{L}$ are exactly the points in $\mathcal{A}$. By including

$\mathcal{L}$ as part of the linear constraints, a MILP can be constructed, and then be solved by optimization solvers such as CPLEX [1] and Gurobi [2], to search differentials with the maximum probability. Similarly, for division property, the DTT of S-Boxes can also be represented as a set of linear (in)equalities and included into a MILP describing the division trails of a cipher.

We propose Algorithm 1 in App. B.2 to search for the minimum number of linear (in)equalities to represent the DDT of an S-Box. For PRESENT S-Box which is used by PHOTON in the inner permutation, we find 8 inequalities (App. B.2) to model the division trails, which is 3 less than [32].[2]

Model MixColumns. The idea is to represent the matrix in bit level. Given the polynomial of the field where the multiplications operate on, the representation of the matrix is unique. For PHOTON permutations based on $\mathbb{F}_{2^4}$, the polynomial is $x^4 + x + 1$.

Then, dummy binary variables are introduced to describe the multiplication with the primitive matrix. Denote $T_{\mathrm{MC}} = (t_{ij})_{n \times n}$, where the binary dummy variables $t_{ij} = 0$ if $m_{ij} = 0$. Then the MixColumns operation of $Y = \mathrm{MC} \times X$, where $X = (x_0, x_1, \cdots, x_{n-1})^T$ and $Y = (y_0, y_1, \cdots, y_{n-1})^T$, can be modeled as $x_j \xrightarrow{\text{copy}} (t_{0j}, t_{1j}, \cdots, t_{(n-1)j})$ and $(t_{i0}, t_{i1}, \cdots, t_{i(n-1)}) \xrightarrow{\text{XOR}} y_i$. For the case of P_{144} for PHOTON-128/16/16, $n = 24$. Then, we can represent MixColumns by linear equations for copy operation and XOR operation. An example of the bit representation of PHOTON permutation P_{144} can be found in B.3.

Objective Function and Rules to Determine the Existence of Zero-Sum. By modeling the operations of ciphers in the above, we are able to describe all the operations in a cipher by linear (in)equalities, and call them the set of linear constraints. In order to construct our MILP, the objective function should be built first. Let's consider a set $\mathbb{X}$ with division property $\mathcal{D}_{\mathbb{K}}^{1,n}$. If $\mathbb{X}$ does not have any zero-sum property, that is the Xor-sum of $\mathbb{X}$ does not balance on any bit, thus we have $\bigoplus_{x \in \mathbb{X}} \pi_{\mathbf{u}}(x)$ is unknown for any unit vector $u \in (\mathbb{F}_2)^n$. Since $\mathbb{X}$ has division property $\mathcal{D}_{\mathbb{K}}^{1,n}$, there must exist a vector $\mathbf{k} \in \mathbb{K}$ such that $u \succeq k$ [3]. Note that $\mathbf{u}$ is a unit vector, thus $\mathbf{u} = \mathbf{k}$, which means $\mathbb{K}$ contains all the n unit vectors. On the other hand, if $\mathbb{K}$ contains all the n unit vectors over $\mathbb{F}_{2^n}$, then for any $0 \neq \mathbf{u} \in (\mathbb{F}_2)^n$ there must exist a unit vector $e \in \mathbb{K}$ such that $u \succeq e$, that is $\bigoplus_{x \in \mathbb{X}} \pi_{\mathbf{u}}(x)$ is unknown. Thus, $\mathbb{X}$ does not have any integral property.

Proposition 1. *[32] Assume $\mathbb{X}$ is a multiset with division property $\mathcal{D}_{\mathbb{K}}^{n}$, then $\mathbb{X}$ does not have zero-sum property if and only if $\mathbb{K}$ contains all the n unit vectors.*

[2] A C/C++ program that verifies our 8 inequalities can cover DDT of PRESENT as the ones given in [32] can be provided if requested. We note that a smaller number of inequalities could help to accelerate searching for zero-sum partitions in some cases (e.g. when the state size is getting large).

[3] Let two vectors $\mathbf{k} = (k_0, k_1, \ldots, k_{m-1})$ and $\mathbf{k}' = (k'_0, k'_1, \ldots, k'_{m-1}) \in \mathbb{Z}^m$, define $k \succeq k'$ if $k_i \geq k'_i$ for all $0 \leq i \leq m-1$; otherwise we denote $k \not\succeq k'$.

Table 3. Number of Rounds of Zero-sums by the MILP Division Property Tool for PHOTON Internal Permutations, in Forward and Backward Direction

Permutation	P_{100}			P_{144}			P_{196}			P_{256}		
	Forward Direction											
#Rounds	4	5	6	4	5	6	4	5	6	4	5	6
[28]	12	20	72	12	24	84	12	24	84	12	28	92
Ours	**11**	20	72	**11**	**23**	84	**11**	24	84	**11**	**27**	92
	Backward Direction											
#Rounds	3	4	5	3	4	5	3	4	5	3	4	5
Ours	**11**	**19***	**71***	**11**	**23**	**83***	**11**	**23***	**83***	**11**	**27**	**91***

* Partial balanced

Thus, we only need to detect whether $\mathbb{K}_r$ contains all unit vectors. In order to check the vectors in $\mathbb{K}_r$, it is equivalent to check the last vectors of all r-round division trails. Denote $(a^0_{n-1}, \cdots, a^0_0) \to \cdots \to (a^r_{n-1}, \cdots, a^r_0)$ an r-round division trail. Thus, we can set the objective function as

$$Min: \; a^r_0 + a^r_1 + \cdots + a^r_{n-1}.$$

Till now, we completely construct the MILP for the division property propagation of a cipher.

Now we are ready to apply this MILP-based division property tool to search for the zero-sums of PHOTON permutations. The zero-sums for variants (both forward and backward directions) with 4-bit S-Box are given in Table 3. Note that we can reach more rounds with a number of texts which is less than the internal state size. The claimed attack complexities by the designers for the above PHOTON permutation variants are 80, 128, 160 and 224 respectively. We only list the ones that help to construct our distinguishers later.

3.2 Observation on the Algebraic Degree Decrease

Denote the input and output of PRESENT S-Box as (x_3, x_2, x_1, x_0) and (y_3, y_2, y_1, y_0), then the ANF of it is as

$$\begin{cases} y_3 = 1 \oplus x_0 \oplus x_1 \oplus x_3 \oplus x_1x_2 \oplus x_0x_1x_2 \oplus x_0x_1x_3 \oplus x_0x_2x_3 \\ y_2 = 1 \oplus x2 \oplus x3 \oplus x_0x_1 \oplus x_0x_3 \oplus x_1x_3 \oplus x_0x_1x_3 \oplus x_0x_2x_3 \\ y_1 = x_1 \oplus x_3 \oplus x_1x_3 \oplus x_2x_3 \oplus x_0x_1x_2 \oplus x_0x_1x_3 \oplus x_0x_2x_3 \\ y_0 = x_0 \oplus x_2 \oplus x_3 \oplus x_1x_2 \end{cases}$$

When x_0 is fixed as constant, then the degree of ANF decreases from 3 to 2 (note that *all the terms of degree 3 contains* x_0). This fact can be used to improve most of the results found by Todo [28], as we will show the details in the following.

3.3 Simple Zero-sum Partitions for PHOTON Permutations

Given the zero-sums of both forward and backward directions of permutations, automatically, one can construct many zero-sums. We take P_{144} for PHOTON-128/16/16 as an example. As we are going to show in next section, since

$$\mathbb{B} \xleftarrow{R^{-4}} \begin{pmatrix} A & C\,C\,C\,C\,C \\ A & C\,C\,C\,C\,C \\ A & C\,C\,C\,C\,C \\ A & C\,C\,C\,C\,C \\ A & C\,C\,C\,C\,C \\ aaac & C\,C\,C\,C\,C \end{pmatrix}, \quad \begin{pmatrix} A\,C\,C\,C\,C & C \\ C\,A\,C\,C\,C & C \\ C\,C\,A\,C\,C & C \\ C\,C\,C\,A\,C & C \\ C\,C\,C\,C\,A & C \\ C\,C\,C\,C\,C & aaac \end{pmatrix} \xrightarrow{R^5} \mathbb{B}$$

where $\mathbb{B}$ *means that the sum is equal to zero in each bit*, it is possible to set up 2^{58} 9-round zero-sum partitions with size 2^{42}, that is

$$\mathbb{B} \xleftarrow{R^{-4}} \begin{pmatrix} A & C\,C\,C\,C & C \\ A & A\,C\,C\,C & C \\ A & C\,A\,C\,C & C \\ A & C\,C\,A\,C & C \\ A & C\,C\,C\,A & C \\ aaac & C\,C\,C\,C & aaac \end{pmatrix} \xrightarrow{R^5} \mathbb{B}.$$

This example allows us to highlight one more time *the possibility to reduce the degree of the S-Box (from 3 to 2) working with input of the form aaac. Our zero-sums exploit this observation to reduce the size of the partitions by* 2^2 *to guarantee the zero-sum property.*

In [8], Boura and Canteaut looked into the the new bound of the concatenated permutation, and add one more round in the middle the zero-sums of single direction, when the non-linear layer is composed of parallel applications of smaller S-boxes. In the next, we apply Subspace trail cryptanalysis to extend one more round in the middle of the zero-sums, and show our applications to PHOTON permutation.

4 Subspace Trails

Subspace Trail Cryptanalysis [18] was recently introduced at FSE 2017. We recall the main concept of such a notation, and refer to [18] for more details. Our treatment here is however meant to be self-contained.

4.1 Subspace Trails of AES-like Permutations

Since PHOTON permutation is an AES-like cipher, in this section we recall the subspace trails of AES presented in [18]. For the following, we only work with vectors and vector spaces over $\mathbb{F}_{2^m}^{n\times n}$ for fixed m, n, and we denote by $\{e_{0,0}, ..., e_{n-1,n-1}\}$ the unit vectors of $\mathbb{F}_{2^m}^{n\times n}$ (e.g. $e_{i,j}$ has a single 1 in row i and column j).

Definition 3. *The column spaces* $\mathcal{C}_i$ *are defined as* $\mathcal{C}_i = \langle e_{0,i}, e_{1,i}, e_{2,i}, ..., e_{n-1,i}\rangle$.

For instance, if $n = 4$ then $\mathcal{C}_0$ corresponds to the symbolic matrix

$$\mathcal{C}_0 = \left\{ \begin{pmatrix} x_0 & 0 & 0 & 0 \\ x_1 & 0 & 0 & 0 \\ x_2 & 0 & 0 & 0 \\ x_3 & 0 & 0 & 0 \end{pmatrix} \middle| \forall x_i \in \mathbb{F}_{2^m}, i = 0, 1, 2, 3 \right\} \equiv \begin{pmatrix} x_0 & 0 & 0 & 0 \\ x_1 & 0 & 0 & 0 \\ x_2 & 0 & 0 & 0 \\ x_3 & 0 & 0 & 0 \end{pmatrix}.$$

Definition 4. *The diagonal spaces $\mathcal{D}_i$ and the inverse-diagonal spaces $\mathcal{ID}_i$ are respectively defined as $\mathcal{D}_i = SR^{-1}(\mathcal{C}_i)$ and $\mathcal{ID}_i = SR(\mathcal{C}_i)$:*

$$\mathcal{D}_i = \langle e_{0,i}, e_{1,(i+1)}, e_{2,(i+2)}, ..., e_{n-1,(i+n-1)} \rangle,$$
$$\mathcal{ID}_i = \langle e_{0,i}, e_{1,(i-1)}, e_{2,(i-2)}, ..., e_{n-1,(i-n+1)} \rangle$$

where all the indexes are taken modulo n.

For instance, if $n = 4$ then $\mathcal{D}_0$ and $\mathcal{ID}_0$ correspond to symbolic matrix

$$\mathcal{D}_0 \equiv \begin{pmatrix} x_0 & 0 & 0 & 0 \\ 0 & x_1 & 0 & 0 \\ 0 & 0 & x_2 & 0 \\ 0 & 0 & 0 & x_3 \end{pmatrix}, \qquad \mathcal{ID}_0 \equiv \begin{pmatrix} x_0 & 0 & 0 & 0 \\ 0 & 0 & 0 & x_1 \\ 0 & 0 & x_2 & 0 \\ 0 & x_3 & 0 & 0 \end{pmatrix}$$

for all $x_i \in \mathbb{F}_{2^m}, i = 0, 1, 2, 3$.

Definition 5. *The i-th mixed spaces $\mathcal{M}_i$ are defined as $\mathcal{M}_i = MC(\mathcal{ID}_i)$.*

For instance, for PHOTON permutation P_{144}, $n = 6$ and $m = 4$ - working in $GF(2^4) \equiv GF(2)[X]/(X^4 + X + 1)$ - then $\mathcal{M}_0$ corresponds to symbolic matrix

$$\mathcal{M}_0 = \begin{pmatrix} 1 & 2 & 8 & 5 & 8 & 2 \\ 2 & 5 & 1 & 2 & 6 & 12 \\ 12 & 9 & 15 & 8 & 8 & 13 \\ 13 & 5 & 11 & 3 & 10 & 1 \\ 1 & 15 & 13 & 14 & 11 & 8 \\ 8 & 2 & 3 & 3 & 2 & 8 \end{pmatrix} \times \mathcal{ID}_0 \equiv \begin{pmatrix} x_0 & 2x_1 & 8x_2 & 5x_3 & 8x_4 & 2x_5 \\ 2x_0 & 12x_1 & 6x_2 & 2x_3 & x_4 & 5x_5 \\ 12x_0 & 13x_1 & 8x_2 & 8x_3 & 15x_4 & 9x_5 \\ 13x_0 & x_1 & 10x_2 & 3x_3 & 11x_4 & 5x_5 \\ x_0 & 8x_1 & 11x_2 & 14x_3 & 13x_4 & 15x_5 \\ 8x_0 & 8x_1 & 2x_2 & 3x_3 & 3x_4 & 2x_5 \end{pmatrix}$$

for all $x_i \in \mathbb{F}_{2^m}, i = 0, 1, \ldots, 5$.

Definition 6. *Let $I \subseteq \{0, 1, ..., n-1\}$. The subspaces $\mathcal{C}_I$, $\mathcal{D}_I$, $\mathcal{ID}_I$ and $\mathcal{M}_I$ are defined as: $\mathcal{C}_I = \bigoplus_{i \in I} \mathcal{C}_i$, $\mathcal{D}_I = \bigoplus_{i \in I} \mathcal{D}_i$, $\mathcal{ID}_I = \bigoplus_{i \in I} \mathcal{ID}_i$, $\mathcal{M}_I = \bigoplus_{i \in I} \mathcal{M}_i$.*

As shown in detail in [18], for any coset $\mathcal{D}_I \oplus a$ there exists unique $b \in \mathcal{C}_I^{\perp}$ such that $R(\mathcal{D}_I \oplus a) = \mathcal{C}_I \oplus b$. Similarly, for any coset $\mathcal{C}_I \oplus a$ there exists unique $b \in \mathcal{M}_I^{\perp}$ such that $R(\mathcal{C}_I \oplus a) = \mathcal{M}_I \oplus b$.

Theorem 1. *For each I and each $a \in \mathcal{D}_I^{\perp}$, there exists one and only one $b \in \mathcal{M}_I^{\perp}$ s.t.*

$$R^2(\mathcal{D}_I \oplus a) = \mathcal{M}_I \oplus b. \tag{1}$$

We refer to [18] for a proof of this statement. We limit to observe that b depends on the initial constant a that defines the coset $\mathcal{D}_I \oplus a$ and on the secret key k.

5 Improved Zero-sum Partitions of P_{144} for PHOTON-128/16/16 and P_{256} for PHOTON-224/32/32

We show how to extend the simple zero-sum partitions in Sect. 3.3 by adding one round in the middle for "free" using the subspace trail cryptanalysis. We emphasize that since this technique is very general, it can be used more generally for any AES-like cipher (as an example, a similar technique allows to explain the 8-round zero-sum partition of AES proposed by Gilbert[4] in [15] starting from the 7-round one proposed in [21]). All the details are only given for the 10-round case. The other cases - 11- and 12-round of P_{144} and all zero-sums of P_{100}, P_{196} and P_{256} - are obtained using the same strategy.

5.1 10-round Zero-sum Partitions for P_{144} of size 2^{47}

In order to set up a 10 round partitioning, we first re-write the simple 9-round zero-sum partition for P_{144} using the subspace trail notation. Since we use the same strategy also for the next zero-sums, we give here all the details. For the following, we define $\mathcal{C}'_I$ and $\mathcal{D}'_I$ for some $I \subseteq \{0, 1, 2, 3\}$ as subspaces of $\mathcal{C}_I$ and $\mathcal{D}_I$ respectively, for which some nibbles are only partially active (i.e. some nibbles can have some active bits and some constant bits). Let $\mathcal{C}'_0$ and $\mathcal{D}'_0$ defined as

$$\mathcal{C}'_0 \equiv \begin{pmatrix} x_0 & 0 & 0 & 0 & 0 & 0 \\ x_1 & 0 & 0 & 0 & 0 & 0 \\ x_2 & 0 & 0 & 0 & 0 & 0 \\ x_3 & 0 & 0 & 0 & 0 & 0 \\ x_4 & 0 & 0 & 0 & 0 & 0 \\ y & 0 & 0 & 0 & 0 & 0 \end{pmatrix}, \qquad \mathcal{D}'_0 \equiv \begin{pmatrix} x_0 & 0 & 0 & 0 & 0 & 0 \\ 0 & x_1 & 0 & 0 & 0 & 0 \\ 0 & 0 & x_2 & 0 & 0 & 0 \\ 0 & 0 & 0 & x_3 & 0 & 0 \\ 0 & 0 & 0 & 0 & x_4 & 0 \\ 0 & 0 & 0 & 0 & 0 & y \end{pmatrix},$$

for all $x_i \in \mathbb{F}_{2^4}, 0 \leq i \leq 4$ and for all $y = 2 \cdot y'$ where y' can take any value in $\mathbb{F}_{2^3}$. It follows that

$$\mathbb{B} \xleftarrow{R^{-4}} \mathcal{D}'_0 \oplus \mathcal{C}'_0 \oplus a \xrightarrow{R^5} \mathbb{B}.$$

How to add one round in the middle for "free"? The idea is *to extend the previous 9-round zero-sum adding one round in the middle*, exploiting the fact that a coset of a column space $\mathcal{C}_I$ is always mapped into a coset of a mixed space $\mathcal{M}_I$ after one round. In more details, using the MILP automatic tool based on division property, one can only found "zero-sum" for which the nibbles can only be active/partial active or constant. This means that other more generic possible cases are not considered, including the one for which some particular (linear) relationships between the nibbles hold. In the following we show how to use subspace trails and the results found by the tool in order to derive these cases. For completeness, we emphasize that the 8-round zero-sum partition proposed by Gilbert [15] - using the super-S-Box view - starting from the 7-round one of

[4] In order to explain such result, Gilbert propose that super-Sbox notation, where *super-Sbox*$(\cdot) :=$ S-Box $\circ ARK \circ MC \circ$ S-Box$(\cdot)$. The same result has been explained in details in [17] using the subspace trail notation.

Knudsen and Rijmen [21] can be derived using the same technique based on the subspace trail notation.

By Theorem 1, it follows that for each constant a there exists a constant b such that

$$R(\mathcal{C}'_0 \oplus a) \subseteq R(\mathcal{C}_0 \oplus a) = M_0 \oplus b,$$

where $\mathcal{C}'_0 \subseteq \mathcal{C}_0$. Before we go on, note that S-Box$(\cdot)$ is a non-linear operation. It follows that while S-Box$(aaaa)$ is well defined (i.e. S-Box$(aaaa) = aaaa$), S-Box$(aaac)$ is not defined in general[5]. Thus, we replace the subspace $\mathcal{C}'_I$ (where some nibbles are only partially active) with the corresponding subspace $\mathcal{C}_I$ (where all the nibbles are only constant or completely active). Note that if the zero-sum property holds for $\mathcal{C}'_I$, it also holds for $\mathcal{C}_I$ since $\mathcal{C}_I \equiv \bigcup_{x\in\mathcal{C}_I\setminus\mathcal{C}'_I} \mathcal{C}'_I \oplus x$ where $\mathcal{C}'_I \subseteq \mathcal{C}_I$. Thus, we introduce $\mathcal{X}$ defined as $\mathcal{X} \equiv \mathcal{D}'_0 \oplus \mathcal{M}_0$ of dimension 47, that is

$$\mathcal{X} = \begin{pmatrix} x_6 & 2x_1 & 8x_2 & 5x_3 & 8x_4 & 2x_5 \\ 2x_0 & x_7 & 6x_2 & 2x_3 & x_4 & 5x_5 \\ 12x_0 & 13x_1 & x_8 & 8x_3 & 15x_4 & 9x_5 \\ 13x_0 & x_1 & 10x_2 & x_9 & 11x_4 & 5x_5 \\ x_0 & 8x_1 & 11x_2 & 14x_3 & x_{10} & 15x_5 \\ 8x_0 & 8x_1 & 2x_2 & 3x_3 & 3x_4 & 2x_5 \oplus y \end{pmatrix}$$

for all $x_i \in \mathbb{F}_{2^4}$ with $0 \leq i \leq 10$ and for all $y = 2 \cdot y'$ as before (where y' can take any value in $\mathbb{F}_{2^3}$). In App. A we present a *generic* formula that allows to compute *directly* the dimension of $\mathcal{X}$ from the dimensions of the initial subspaces $\mathcal{D}'_I$ and $\mathcal{C}'_J$.

10-round zero-sums. Since for each constant b $\mathcal{X} \oplus b = \bigcup_{c\in\mathcal{D}'_0} \mathcal{M}_0 \oplus (b \oplus c) = \bigcup_{d\in\mathcal{M}_0} \mathcal{D}'_0 \oplus (b \oplus d)$, it follows that

$$\mathbb{B} \xleftarrow{R^{-4}} \bigcup_c \mathcal{C}_0 \oplus c \xleftarrow{R^{-1}} \mathcal{X} \oplus b \equiv \bigcup_{d\in\mathcal{M}_0} \mathcal{D}'_0 \oplus (b \oplus d) \xrightarrow{R^5} \mathbb{B}.$$

As a result, starting in the middle with a coset of $\mathcal{X}$ implies zero-sum after 5-round decryption/encryption, that is $\mathbb{B} \xleftarrow{R^{-5}} \mathcal{X} \oplus b \xrightarrow{R^5} \mathbb{B}$. Thus for partitions in $\mathcal{X} \oplus b$ of size 2^{47}, we construct 10-round zero-sum partition for P_{144}. Note that this complexity is significantly below the birthday bound of the security level 2^{64}.

5.2 11-round Zero-sum Partitions for P_{144} of size 2^{107}

As before, we first present the simple10-round zero-sums found by the tool:

$$\mathbb{B} \xleftarrow{R^{-5}} \begin{pmatrix} A & A & A & C & C & C \\ A & A & A & C & C & C \\ A & A & A & C & C & C \\ A & A & A & A & C & C \\ A & A & A & A & C & C \\ A & A & A & A & C & C \end{pmatrix}, \begin{pmatrix} A & C & C & C & C & C \\ C & A & C & C & C & C \\ C & C & A & C & C & C \\ C & C & C & A & C & C \\ C & C & C & C & A & C \\ C & C & C & C & C & ccca \end{pmatrix} \xrightarrow{R^5} \mathbb{B}$$

[5] More precisely, S-Box$(aaac)$ is a subset of 8 elements of $\{0x0, 0x1, ..., 0xf\}$. On the other hand, such subset depends on the details of the S-Box function and doesn't have any particular property.

Let $\mathcal{D}'_0$ defined as before, and let

$$\mathcal{C}_{0,1,2} \oplus (\mathcal{C}_3 \cap \mathcal{D}_{0,1,5}) = \begin{pmatrix} x_0 & x_6 & x_{12} & 0 & 0 & 0 \\ x_1 & x_7 & x_{13} & 0 & 0 & 0 \\ x_2 & x_8 & x_{14} & 0 & 0 & 0 \\ x_3 & x_9 & x_{15} & x_{18} & 0 & 0 \\ x_4 & x_{10} & x_{16} & x_{19} & 0 & 0 \\ x_5 & x_{11} & x_{17} & x_{20} & 0 & 0 \end{pmatrix}, \ \forall x_i \in \mathbb{F}_{2^4}, 0 \le i \le 20.$$

Thus, for each constant a:

$$\mathbb{B} \xleftarrow{R^{-5}} \mathcal{D}'_0 \oplus \mathcal{C}_{0,1,2} \oplus (\mathcal{C}_3 \cap \mathcal{D}_{0,1,5}) \oplus a \xrightarrow{R^5} \mathbb{B}.$$

This size of these 10-round zero-sums found by the tool is 2^{89}, which is much higher than the one proposed in the previous section obtained by extending in the middle of a 9-round exploiting the subspace trail. Similar to before, for each constant a there exists unique b such that after one round encryption

$$R(\mathcal{C}_{0,1,2} \oplus (\mathcal{C}_3 \cap \mathcal{D}_{0,1,5}) \oplus a) = \mathcal{M}_{0,1,2} \oplus (\mathcal{M}_3 \cap \mathcal{C}_{0,1,5}) \oplus b.$$

Let $\mathcal{X}$ defined as $\mathcal{X} \equiv \mathcal{D}'_0 \oplus \mathcal{M}_{0,1,2} \oplus (\mathcal{M}_3 \cap \mathcal{C}_{0,1,5})$ of dimension 107. By similar argumentation as before, for each b, we have

$$\mathbb{B} \xleftarrow{R^{-6}} \mathcal{X} \oplus b \xrightarrow{R^5} \mathbb{B}.$$

Thus, one can construct 11-round zero-sum partitions of size 2^{107} for P_{144}.

5.3 12-round Zero-sum Partitions of P_{144}

Impossibility to set up a 12-round zero-sum partition with full balance. By tool, the best result (in term of minimum number of active bits) that we obtained for 11-round P_{144} is given by

$$\mathbb{B} \xleftarrow{R^{-5}} \begin{pmatrix} A & A & A & C & C & C \\ A & A & A & C & C & C \\ A & A & A & C & C & C \\ A & A & A & A & C & C \\ A & A & A & A & C & C \\ A & A & A & A & C & C \end{pmatrix}, \begin{pmatrix} A & C & C & C & A & A \\ A & A & C & C & C & A \\ A & A & A & C & C & C \\ A & A & A & A & C & C \\ C & A & A & A & A & C \\ C & C & A & A & A & A \end{pmatrix} \xrightarrow{R^6} \mathbb{B}.$$

Thus, it is possible to construct a 11-round zero-sums, that is

$$\mathbb{B} \xleftarrow{R^{-5}} \begin{pmatrix} A & A & A & C & A & A \\ A & A & A & C & C & A \\ A & A & A & C & C & C \\ A & A & A & A & C & C \\ A & A & A & A & A & C \\ A & A & A & A & A & A \end{pmatrix} \xrightarrow{R^6} \mathbb{B}.$$

Also in this case, we can re-write these zero-sums using the subspace trail notation. In particular, denoted by $\mathcal{Z} \equiv \big[\mathcal{D}_{0,1,2} \oplus (\mathcal{D}_3 \cap \mathcal{C}_{0,1,2})\big] \oplus \big[\mathcal{C}_{0,1,2} \oplus (\mathcal{C}_3 \cap \mathcal{D}_{0,1,5})\big]$, it follows that $\mathbb{B} \xleftarrow{R^{-5}} \mathcal{Z} \xrightarrow{R^6} \mathbb{B}$.

In the same way as before, one can try to extend these zero-sums in the middle. However, the dimension of $\mathcal{X} \equiv R(\mathcal{C}) \cap \mathcal{D}$ in this case is equal to $dim = 144$ (we refer to App. A for all the details). It follows that this is an example for which it is not possible to set up a 12-round zero-sum partition starting from a 11-round one and using the strategy just presented.

12-round *partial* zero-sum partition for P_{144} of size 2^{127}. By tool, we found the following 12-round *partial* zero-sum partition of P_{144} of size 2^{127}:

$$\mathbb{PB} \xleftarrow{R^{-6}} \begin{pmatrix} A\,A\,A\,A\,A & C \\ A\,A\,A\,A\,A & C \\ A\,A\,A\,A\,A & C \\ A\,A\,A\,A\,A & C \\ A\,A\,A\,A\,A & aaac \\ A\,A\,A\,A\,A & A \end{pmatrix} \xrightarrow{R^{6}} \mathbb{B}.$$

5.4 Full-Round Zero-sum Partitions of P_{256} for PHOTON-224/32/32

In this subsection, we propose a full-round zero-sum of P_{256} for PHOTON-224/32/32. Let's start with the following 11-round zero-sum partition found by the tool

$$\mathbb{PB} \xleftarrow{R^{-5}} \begin{pmatrix} A\,A\,A\,C\,C\,C\,C\,C \\ A\,A\,A\,C\,C\,C\,C\,C \\ A\,A\,A\,C\,C\,C\,C\,C \\ A\,A\,A\,C\,C\,C\,C\,C \\ A\,A\,A\,C\,C\,C\,C\,C \\ A\,A\,A\,C\,C\,C\,C\,C \\ A\,A\,A\,C\,C\,C\,C\,C \\ A\,A\,C\,C\,C\,C\,C\,C \end{pmatrix}, \begin{pmatrix} A\,C\,C\,C\,C\,C\,A\,A \\ A\,A\,C\,C\,C\,C\,C\,C \\ A\,A\,A\,C\,C\,C\,C\,C \\ C\,A\,A\,A\,C\,C\,C\,C \\ C\,C\,A\,A\,A\,C\,C\,C \\ C\,C\,C\,A\,A\,A\,C\,C \\ C\,C\,C\,C\,A\,A\,A\,C \\ C\,C\,C\,C\,C\,A\,A\,A \end{pmatrix} \xrightarrow{R^{6}} \mathbb{B}$$

which can be rewritten using the subspace trail notation as $\mathbb{B} \xleftarrow{R^{-5}} \mathcal{Z} \oplus a \xrightarrow{R^{6}} \mathbb{B}$ for each constant a, where $\mathcal{Z} \equiv \big[\mathcal{C}_{0,1} \oplus (\mathcal{C}_2 \cap \mathcal{D}_{0,1,3,4,5,6,7})\big] \oplus \big[\mathcal{D}_{0,7} \oplus (\mathcal{D}_6 \cap \mathcal{C}_{0,1,2,3,4,5,6})\big]$. Using the subspace trail cryptanalysis (see Sect. 5.1 for details), let the space $\mathcal{X}$ defined as

$$\mathcal{X} \equiv \big[\mathcal{M}_{0,1} \oplus (\mathcal{M}_2 \cap \mathcal{C}_{0,1,3,4,5,6,7})\big] \oplus \big[\mathcal{D}_{0,7} \oplus (\mathcal{D}_6 \cap \mathcal{C}_{0,1,2,3,4,5,6})\big]$$

of dimension 184. Since for each b:

$$\mathbb{B} \xleftarrow{R^{-6}} \mathcal{X} \oplus b \xrightarrow{R^{6}} \mathbb{B},$$

it is possible to set up full-round zero-sum partitions for P_{256}.

6 Zero-sum Partitions for the PHOTON Hash Functions

The utilization of degrees of freedom has always been one of the most powerful cryptanalyst tool for sponge-like hash functions, thus reducing this ability as much as possible greatly increases the confidence in the sponge-like hash function's security. For PHOTON, this "small-r" sponge-like shape makes the amount of freedom degrees available at the input of each internal permutation call during the absorbing phase is extremely small. Thus, even though we manage to find the full-round distinguishers for the internal permutation, the amount of freedom degrees is so thin that utilizing this flaw will not threaten the security of PHOTON as a hash function.

In this section, we explain our results for hash function PHOTON. Following the sponge strategy, at iteration i PHOTON absorbs the message block m_i on leftmost part of the internal state S_i, and then applies the permutation P_t. Following that is the squeezing phase. We take PHOTON-160/36/36 as the example, and the absorbing positions of the state array are underlined as below

$$\begin{pmatrix} \underline{s_{0,0}} & \underline{s_{0,1}} & \underline{s_{0,2}} & \underline{s_{0,3}} & \underline{s_{0,4}} & \underline{s_{0,5}} & \underline{s_{0,6}} \\ \underline{s_{1,0}} & \underline{s_{1,1}} & s_{1,2} & s_{1,3} & s_{1,4} & s_{1,5} & s_{1,6} \\ s_{2,0} & s_{2,1} & s_{2,2} & s_{2,3} & s_{2,4} & s_{2,5} & s_{2,6} \\ s_{3,0} & s_{3,1} & s_{3,2} & s_{3,3} & s_{3,4} & s_{3,5} & s_{3,6} \\ s_{4,0} & s_{4,1} & s_{4,2} & s_{4,3} & s_{4,4} & s_{4,5} & s_{4,6} \\ s_{5,0} & s_{5,1} & s_{5,2} & s_{5,3} & s_{5,4} & s_{5,5} & s_{5,6} \\ s_{6,0} & s_{6,1} & s_{6,2} & s_{6,3} & s_{6,4} & s_{6,5} & s_{6,6} \end{pmatrix}$$

With data of size 2^{20}, we can find a 4-round zero-sum partition for PHOTON-160/36/36.

$$\begin{pmatrix} A & A & A & A & A & C & C \\ C & C & C & C & C & C & C \\ C & C & C & C & C & C & C \\ C & C & C & C & C & C & C \\ C & C & C & C & C & C & C \\ C & C & C & C & C & C & C \\ C & C & C & C & C & C & C \end{pmatrix} \xrightarrow{R^4} \mathbb{B}$$

7 Comparison with Generic Approaches

A natural question to ask here is how generic approaches to construct zero-sums or zero-sum partitions compare with our dedicated approach for PHOTON. Here we tackle this question, considering as starting point the zero-sum results on Keccak.

We first briefly recall the generic method for constructing a zero-sum structure which is inspired by the attack against XHASH in [4] (brought to attention of Keccak Team [5] by Jean-Philippe Aumasson). The strategy is the following. Assume we are looking for a set $\mathcal{Z} = \{z_i\}$ of N elements in $\mathbb{F}_{2^n}$ such that $\bigoplus_i z_i = \bigoplus_i f(z_i) = 0$. As first step, one considers N random value $x_i \in \mathbb{F}_{2^n}$ and computes $\mathcal{X} = \{x_i || f(x_i)\}_i$ where $x_i || f(x_i) \in \mathbb{F}_{2^{2n}}$. Let

$$A = \bigoplus_{\mathcal{X}} x_i || f(x_i) \equiv \bigoplus_{\mathcal{X}} x_i || \bigoplus_{\mathcal{X}} f(x_i).$$

If A is equal to zero (prob. 2^{-2n}), then the problem is solved. Assume $A \neq 0$. The idea is to consider other M random elements - for a certain M - $y_i \in \mathbb{F}_{2^n}$ and compute $\{y_i || f(y_i)\}_i$. Then, one computes binary coefficients $\{a_i\}_{i=0,\ldots,M-1}$

that satisfy the following equality

$$\bigoplus_{i=0}^{M-1} a_i \cdot (x_i \oplus y_i || f(x_i) \oplus f(y_i)) = A.$$

Observe that such condition is satisfied with non-negligible property if $M > 2n$ - in particular[6], it is satisfied with probability higher than 99.99% if $M = 2n+10$. Assume that a solution of the previous equality is found. The set $\mathcal{Z} = \{z_i\}$ is defined as

$$z_i \equiv \begin{cases} a_i \cdot y_i \oplus (1 \oplus a_i) \cdot x_i, & \text{if } i \leq M \\ x_i, & \text{if } i > M \end{cases}$$

Such set provided a solution of the problem, and the total cost of this algorithm is well approximated by $N + 2n + 10 \approx N$ computations/encryptions.

For a zero-sum distinguisher, in order to construct the set $\mathcal{Z}$, one needs to start from some intermediate state and compute forward and backward to get the input and output of the zero-sum. Since for our zero-sums for PHOTON permutation, the number of forward and backward rounds are almost equal, we assume the computations equivalent of $N/2$ calls to the permutations. Also we assume the cost of encryption and decryption are the same. For the values of partition size N given in Table 1, our method for generating zero-sum structures as distinguishers for PHOTON, is more efficient than the generic method by a factor 2. For instance, consider the case of P_{256} for PHOTON-224/32/32, the complexity of generating our full round distinguisher needs $6/12 \times 2^{184} = 2^{183}$ encryptions while for the generic method it is 2^{184} encryptions.

Remarks. There are a number of related zero-sum results in the literature, most prominently perhaps a full-round result on Keccak (NIST SHA-3) [3,8,14]. In all these works, the computational complexity difference between zero-sum method and the generic method is usually very small (a factor 2). Even if a distinguisher can be considered meaningful only if this difference is significant, the Keccak Team published a note [5] where they confirmed the validity of such distinguishers: "*[...] the zero-sum distinguishers of [3,8] are valid, albeit with a very small advantage*". Our approach to set up zero-sum distinguishers on PHOTON permutation follows the same philosophy.

Before we go on, one may ask the implication of a zero-sum distinguisher. Indeed, even if it provides a way to distinguish a permutation from a random one, for a hash function it is still difficult that the distinguisher can be used to set up an attack. To give a concrete example of implications of a zero-sum distinguisher, we recall its implication on Keccak (Sect. 4 of [5]). The Keccak Team claimed it is very unlikely that the zero-sum distinguishers can result in actual attacks against Keccak calling (reduced-round) versions of Keccak-f, but

[6] Given a fixed set $\{a_i\}_i$, they satisfy the required equality with probability 2^{-2n}. It follows that given $2n + \varepsilon$ sets, at least one of them satisfy it with probability $1 - (1 - 2^{-2n})^{2n+\varepsilon} \approx 1 - e^{\varepsilon}$, assuming $2n \gg 1$. For a probability of success higher than 99.99%, it follows $\varepsilon \geq 10$.

still they confirmed the distinguishers described in [3,8] show non-ideal properties of the (reduced round) Keccak-f permutation and they decided to increase the number of rounds (e.g., for Keccak-f [1600] from 18 to 24 rounds) in round 2 of the SHA-3 competition. Since PHOTON follows exactly the classical sponge strategy, we would believe our full-round zero-sum distinguishers for PHOTON permutations have similar implications on the family of hash functions to the ones on Keccak hash function.

Finally, we emphasize that such distinguishers based on zero-sum cannot be considered meaningless because they cannot be set up for any arbitrary number of rounds. In other words, the inside-out approach used in this paper and in literature to set up distinguishers doesn't work for any arbitrary number of rounds of the inner permutation of the Sponge function. For example, it is not possible to set up a zero-sum distinguisher in the case of PHOTON-80/20/16 with 12-round, while it is possible for the other cases. In these last cases and in the same way, if the number of rounds of PHOTON inner permutations are increased from 12 to (e.g.) 16, our zero-sum distinguishers proposed in this paper can not cover the full inner permutation, which then becomes indistinguishable from a pseudo-random permutation as in the assumptions/requirements provided by the sponge construction/design.

8 Conclusions

We presented zero-sum-related properties of the full-round permutation of many members of the PHOTON family of hash functions. Observations on the used S-Box, a tool-approach to finding division trails, and an inside-out approach with a technique to add a round in the middle are important ingredients. This seems to be the first time that the individual techniques that we employ are used in combination, and the result is on the full version of an ISO standard.

Our results are theoretical in nature and we stress that *there is currently no reason to believe that the security of PHOTON as a hash function is endangered.* It will be interesting to see applications of our approach to other constructions.

Acknowledgements. The authors would like to thank Meicheng Liu and Jian Guo for their fruitful discussions, and the anonymous reviewers for their comments and suggestions.

References

1. http://www.ibm.com/software/integration/optimization/cplex-optimizer/.
2. http://www.gurobi.com/.
3. J.-P. Aumasson and W. Meier, "Zero-sum distinguishers for reduced Keccak-f and for the core functions of Luffa and Hamsi," presented at the Rump Session of Cryptographic Hardware and Embedded Systems - CHES 2009, https://131002.net/data/papers/AM09.pdf.

4. Bellare, Mihir and Micciancio, Daniele, “A New Paradigm for Collision-Free Hashing: Incrementality at Reduced Cost,” in *EUROCRYPT 1997*, ser. LNCS, vol. 1233, 1997, pp. 163–192.
5. G. Bertoni, J. Daemen, M. Peeters, and G. Van Assche, “Note on zero-sum distinguishers of Keccak-f,” http://keccak.noekeon.org/NoteZeroSum.pdf.
6. ——, “Sponge functions,” Ecrypt Hash Workshop, 2007.
7. A. Biryukov and A. Shamir, “Structural cryptanalysis of SASAS,” in *EUROCRYPT 2001*, ser. LNCS, vol. 2045, 2001, pp. 394–405.
8. C. Boura and A. Canteaut, “A zero-sum property for the KECCAK-f permutation with 18 rounds,” in *IEEE International Symposium on Information Theory, ISIT 2010, June 13-18, 2010, Austin, Texas, USA, Proceedings.* IEEE, 2010, pp. 2488–2492. [Online]. Available: http://dx.doi.org/10.1109/ISIT.2010.5513442
9. ——, “Another View of the Division Property,” in *CRYPTO 2016*, ser. LNCS, vol. 9814, 2016, pp. 654–682.
10. ——, “Another View of the Division Property,” in *CRYPTO 2016*, ser. LNCS, vol. 9814, 2016.
11. J. Daemen, L. R. Knudsen, and V. Rijmen, “The Block Cipher Square,” in *FSE 1997*, ser. LNCS, vol. 1267, 1997, pp. 149–165.
12. H. Dobbertin, “Cryptanalysis of MD5 compress,” 1996, presented at the rump session of Eurocrypt 1996.
13. ——, “The Status of MD5 After a Recent Attack,” *CryptoBytes*, vol. 2, no. 2, 1996, ftp://ftp.rsasecurity.com/pub/cryptobytes/crypto2n2.pdf.
14. M. Duan and X. Lai, “Improved zero-sum distinguisher for full round Keccak-f permutation,” *Chinese Science Bulletin*, vol. 57, no. 6, pp. 694–697, 2012.
15. H. Gilbert, “A Simplified Representation of AES,” in *ASIACRYPT 2014*, ser. LNCS, vol. 8873, 2014, pp. 200–222.
16. H. Gilbert and M. Minier, “A collision attack on 7 rounds of Rijndael,” in *AES Candidate Conference*, 2000, pp. 230–241.
17. L. Grassi and C. Rechberger, “New and Old Limits for AES Known-Key Distinguishers,” Cryptology ePrint Archive, Report 2017/255, 2017. [Online]. Available: http://eprint.iacr.org/2017/255
18. L. Grassi, C. Rechberger, and S. Rønjom, “Subspace Trail Cryptanalysis and its Applications to AES,” *IACR Transactions on Symmetric Cryptology*, vol. 2016, no. 2, pp. 192–225, 2017. [Online]. Available: http://tosc.iacr.org/index.php/ToSC/article/view/571
19. J. Guo, T. Peyrin, and A. Poschmann, “The PHOTON Family of Lightweight Hash Functions,” in *CRYPTO 2011*, ser. LNCS, vol. 6841, 2011, pp. 222–239.
20. J. Jean, M. Naya-Plasencia, and T. Peyrin, “Improved Rebound Attack on the Finalist Grøstl,” in *FSE 2012*, ser. LNCS, vol. 9056, 2012, pp. 110–126.
21. L. R. Knudsen and V. Rijmen, “Known-Key Distinguishers for Some Block Ciphers,” in *ASIACRYPT 2007*, ser. LNCS, vol. 4833, 2007, pp. 315–324.
22. L. R. Knudsen and D. Wagner, “Integral cryptanalysis,” in *FSE 2002*, ser. LNCS, vol. 2365. Springer, 2002, pp. 112–127.
23. Knudsen, Lars R., “Truncated and higher order differentials,” in *FSE 1994*, ser. LNCS, vol. 1008, 1995, pp. 196–211.
24. S. Lucks, “Attacking seven rounds of Rijndael under 192-bit and 256-bit keys,” in *AES Candidate Conference*, 2000, pp. 215–229.
25. L. Sun, W. Wang, and M. Wang, “MILP-Aided Bit-Based Division Property for Primitives with Non-Bit-Permutation Linear Layers,” Cryptology ePrint Archive, Report 2016/811, 2016, http://eprint.iacr.org/2016/811.

26. S. Sun, L. Hu, P. Wang, K. Qiao, X. Ma, and L. Song, "Automatic Security Evaluation and (Related-key) Differential Characteristic Search: Application to SIMON, PRESENT, LBlock, DES(L) and Other Bit-Oriented Block Ciphers," in *ASIACRYPT 2014*, ser. LNCS, vol. 8873, 2014, pp. 158–178.
27. Y. Todo, "Integral cryptanalysis on full MISTY1," in *CRYPTO 2015, Part I*, ser. LNCS, vol. 9215, 2015, pp. 413–432.
28. ——, "Structural Evaluation by Generalized Integral Property," in *EUROCRYPT 2015*, ser. LNCS, vol. 9056, 2015, pp. 287–314.
29. Y. Todo and M. Morii, "Bit-Based Division Property and Application to Simon Family," in *FSE 2016*, ser. LNCS, vol. 9783, 2016, pp. 357–377.
30. D. Wagner, "The boomerang attack," in *FSE 1999*, ser. LNCS, vol. 1636, 1999, pp. 156–170.
31. Q. Wang, Z. Liu, K. Varici, Y. Sasaki, V. Rijmen, and Y. Todo, "Cryptanalysis of reduced-round SIMON32 and SIMON48," in *INDOCRYPT 2014*, ser. LNCS, vol. 8885, 2014, pp. 143–160.
32. Z. Xiang, W. Zhang, Z. Bao, and D. Lin, "Applying MILP Method to Searching Integral Distinguishers Based on Division Property for 6 Lightweight Block Ciphers," in *ASIACRYPT 2016*, ser. LNCS, vol. 10031, 2016, pp. 648–678.
33. W. Zhang and V. Rijmen, "Division Cryptanalysis of Block Ciphers with a Binary Diffusion Layer," Cryptology ePrint Archive, Report 2017/188, 2017, http://eprint.iacr.org/2017/188.

A How to Compute *Directly* Dimension of $\mathcal{X}$ in Sect. 5?

In Sect. 5, it is possible to compute the dimension dim of $\mathcal{X}$ without writing the explicit matrix. Here we give all the details.

Assume we are working with $n \times n$ matrix, and consider two input spaces: a diagonal space $\mathcal{D}$ with dimension n_d and a columns space $\mathcal{C}$ with dimension n_c - here the dimension denotes the number of non-constant bits. Due to the argumentation given in Sect. 5 (e.g. see footnote 5), each nibble of $\mathcal{C}$ can be only active or constant (i.e. it is not possible that some bits of a nibble are active and others are constant). It follows that n_c must be a multiple of 4, that is $n_c \leftarrow n_c + (n_c \mod 4)$. Moreover, let n_D the number of diagonals of $\mathcal{D}$ with at least one non-constant nibble, and let n_C the number of columns of $\mathcal{C}$ with at least one non-constant nibble.

Since $\mathcal{X}$ are defined as $R(\mathcal{C}) \oplus \mathcal{D}$, in order to compute the dimension of $\mathcal{Y}$ or $\mathcal{X}$ we consider separately the two cases, that is (1) $n_D + n_C < n$ and (2) $n_D + n_C \geq n$.

Case (1): $n_D + n_C < n$. Let $\mathcal{M} \equiv R(\mathcal{C})$, where the dimension n_m - i.e. of non-constant and *independent* variables - number of the mixed space $\mathcal{M}$ is equal to n_c. In this case, since $\mathcal{M} \cap \mathcal{D} = \{0\}$ (see [18] for details), it follows that the dimension dim of $\mathcal{X}$ or $\mathcal{Y}$ is simply defined as: $dim = n_d + n_c$. As an example, for $\mathcal{X}$ on 10-round we have that $n_D = 1$, $n_C = 2$, $n_d = 23$ and $n_c = 34$, while for $\mathcal{X}$ on 11-round we have that $n_D = 1$, $n_C = 3$, $n_d = 23$ and $n_c = 83$.

Case (2): $n_D + n_C \geq n$. In this case, dim is given by $dim = n_d + n_c - dim(\mathcal{M} \cap \mathcal{D})$ since $\mathcal{M} \cap \mathcal{D} \neq \{0\}$. In order to compute "quickly" $dim(\mathcal{M} \cap \mathcal{D})$, one can use the following analysis. For the following, we limit to consider for simplicity the case in which a nibble can be active or constant (in other words, it is not possible that some bits of a given nibble are active, and the others constant).

Let n_D^i the number of active nibbles in the i-th column for $i = 0, ..., n-1$ of $\mathcal{D}$, and similarly let n_C^i the number of active nibbles in the i-th column for $i = 0, ..., n-1$ of $SR(\mathcal{C}) = MC^{-1}(\mathcal{M})$, i.e. of independent variables of the i-th column of $\mathcal{M}$. Obviously, $n_D = \sum_i n_D^i$ and $n_C = \sum_i n_C^i$. It follows that $dim = 4 \cdot \sum_{i=0}^{n-1} \min(n_D^i + n_C^i, n)$.

As example, we propose in details the calculation to compute the dimension of $\mathcal{X}$ defined in Sect. 5.3. Let $\mathcal{D} = \mathcal{D}_{0,1,2} \oplus (\mathcal{D}_3 \cap \mathcal{C}_{0,1,2})$ with $n_d = 84$ and $n_D = 4$ and $\mathcal{C} = \mathcal{C}_{0,1,2} \oplus (\mathcal{C}_3 \cap \mathcal{D}_{0,1,5})$ with $n_c = 84$ and $n_C = 4$. Since $n_D + n_C \geq 6$, the intersection $R(\mathcal{C}) \cap \mathcal{D}$ is not null. Using the previous formula, we have

$$(n_D^0, n_D^1, n_D^2, n_D^3, n_D^4, n_D^5) = (4, 4, 4, 3, 3, 3)$$
$$(n_C^0, n_C^1, n_C^2, n_C^3, n_C^4, n_C^5) = (4, 3, 3, 3, 4, 4).$$

It follows that the dimension of $R(\mathcal{C}) \cap \mathcal{D}$ is equal to $dim = 144$.

B Definition of Division Property and Its MILP Model of Operations

B.1 Definition of Division Property

We first briefly recall the definition related to division property here:
For any $a \in \mathbb{F}_{2^n}$, let $a[i]$ denote the i-th bit of a, and the Hamming weight of a is defined as $w(a) = \sum_{i=0}^{n-1} a[i]$. For any $\mathbf{a} = (a_0, a_1, \ldots, a_{m-1}) \in (\mathbb{F}_{2^n})^m$, the vectorial Hamming weight of $\mathbf{a}$ is defined as $w(\mathbf{a}) = (w(a_0), w(a_1), \ldots, w(a_{m-1}))$ where $w(a_i)$ is the Hamming weight of a_i.

Definition 7 (Bit Product Function). *For any $u \in \mathbb{F}_{2^n}$, let $\pi_u(x)$ be a function from $\mathbb{F}_{2^n}$ to $\mathbb{F}_2$. For any $x \in \mathbb{F}_{2^n}$, define $\pi_u(x)$ as $\pi_u(x) = \prod_{i=0}^{n-1} x[i]^{u[i]}$. For any $\mathbf{u} \in (\mathbb{F}_{2^n})^m$, let $\pi_{\mathbf{u}}(x)$ be a function from $(\mathbb{F}_{2^n})^m$ to $\mathbb{F}_2$. For all $\mathbf{u} = (u_0, u_1, \ldots, u_{m-1})$ and $x = (x_0, x_1, \ldots, x_{m-1}) \in (\mathbb{F}_{2^n})^m$, define $\pi_{\mathbf{u}}(x)$ as $\pi_{\mathbf{u}}(x) = \prod_{i=0}^{m-1} \pi_{u_i}(x_i)$.*

Definition 8 (Division Property [28]). *Let $\mathbb{X}$ be a multiset whose elements take a value of $(\mathbb{F}_{2^n})^m$, and k be an m-dimensional vector whose coordinates take values between 0 and n. When the multiset $\mathbb{X}$ has the division property $\mathcal{D}^{n,m}_{\mathbf{k}^{(0)},\mathbf{k}^{(1)},\ldots,\mathbf{k}^{(q-1)}}$, it fulfills the following conditions: The parity of $\pi_{\mathbf{u}}(x)$ over all $x \in \mathbb{X}$ is always even when*

$$\{\mathbf{u} = (u_0, u_1, \ldots, u_{m-1}) \in (\mathbb{F}_{2^n})^m | w(u) \not\succeq \mathbf{k}^{(0)}, \ldots, w(u) \not\succeq \mathbf{k}^{(q-1)}\}.$$

Definition 9 (Division Trail [32]). *Let R denote the round function of an iterated block cipher. Assume the input multiset to the block cipher has initial division property $\mathbb{K}_0 = \mathcal{D}^{n,m}_{\mathbf{k}}$, and denote the division property after i-round through R by $\mathcal{D}^{n,m}_{\mathbb{K}_i}$. We have the following trail of division property propagations:*

$$\mathbf{k} \equiv \mathbb{K}_\mathbf{0} \xrightarrow{\mathbf{R}} \mathbb{K}_\mathbf{1} \xrightarrow{\mathbf{R}} \cdots \xrightarrow{\mathbf{R}} \mathbb{K}_\mathbf{i}.$$

For $(\mathbf{k_0}, \mathbf{k_1}, \cdots \mathbf{k_r})$, if $\mathbf{k_i}$ can propagate to $\mathbf{k_{i+1}}$, for all $0 \le i \le r-1$, we call $(\mathbf{k_0}, \mathbf{k_1}, \cdots \mathbf{k_r})$ an r-round division trail.

B.2 Model Division Propagation of PRESENT S-Box

For an n-bit S-Box, assume the input multiset $\mathbb{X}$ takes values $(x_{n-1}, \ldots, x_1, x_0) \in \mathbb{F}_{2^n}$, and the output multiset $\mathbb{Y}$ takes values $(y_{n-1}, \ldots, y_1, y_0) \in \mathbb{F}_{2^n}$, i.e. $(x_{n-1}, \ldots, x_1, x_0) \xrightarrow{\text{S-Box}} (y_{n-1}, \ldots, y_1, y_0)$. Assume the input and output division property of S-Box is $\mathcal{D}^n_{(a_{n-1},\ldots,a_1,a_0)}$ and $\mathcal{D}^n_{(b_{n-1},\ldots,b_1,b_0)}$ respectively, i.e. $\mathcal{D}^n_{(a_{n-1},\ldots,a_1,a_0)} \xrightarrow{\text{S-Box}} \mathcal{D}^n_{(b_{n-1},\ldots,b_1,b_0)}$.

The DDT of the PRESENT S-Box is given as below, where the left column is value of $(a_{n-1}, \ldots, a_1, a_0)$ and the right column is all the possible values of $(b_{n-1}, \ldots, b_1, b_0)$. There are 47 division trails for PRESENT S-Box, as showed in details in Tab. 4. Given the DDT of a n-bit S-Box, we denote the set of the impossible

Table 4. Division Trail Table of PRESENT S-Box

Input $\mathcal{D}^4_{\mathbf{k}}$	Output $\mathcal{D}^4_{\mathbb{K}}$
(0,0,0,0)	(0,0,0,0)
(0,0,0,1)	(0,0,0,1) (0,0,1,0) (0,1,0,0) (1,0,0,0)
(0,0,1,0)	(0,0,0,1) (0,0,1,0) (0,1,0,0) (1,0,0,0)
(0,0,1,1)	(0,0,1,0) (0,1,0,0) (1,0,0,0)
(0,1,0,0)	(0,0,0,1) (0,0,1,0) (0,1,0,0) (1,0,0,0)
(0,1,0,1)	(0,0,1,0) (0,1,0,0) (1,0,0,0)
(0,1,1,0)	(0,0,0,1) (0,0,1,0) (1,0,0,0)
(0,1,1,1)	(0,0,1,0) (1,0,0,0)
(1,0,0,0)	(0,0,0,1) (0,0,1,0) (0,1,0,0) (1,0,0,0)
(1,0,0,1)	(0,0,1,0) (0,1,0,0) (1,0,0,0)
(1,0,1,0)	(0,0,1,0) (0,1,0,0) (1,0,0,0)
(1,0,1,1)	(0,0,1,0) (0,1,0,0) (1,0,0,0)
(1,1,0,0)	(0,0,1,0) (0,1,0,0) (1,0,0,0)
(1,1,0,1)	(0,0,1,0) (0,1,0,0) (1,0,0,0)
(1,1,1,0)	(0,1,0,1) (1,0,1,1) (1,1,1,0)
(1,1,1,1)	(1,1,1,1)

division trails as *Bset*, and the size of it as *trailsize*. For PRESENT, $trailsize = 256 - 47 = 209$. We denote the total number of linear inequalities in the H-Representation

Algorithm 1 Search minimum amount of linear inequalities representing DDT of S-Box

```
/* $len$ is the total number of linear inequalities in H-Represent of DDT, and $Coefset$ is the coefficients of these inequalities; $Bset$ is the set of impossible division trails need to be removed, and $trailsize$ is the total number of these trails $path$ is the system of minimum amount of linear inequalities describing DDT */

Input: ($Coefset$, $Bset$)
Output: $path$

  /* Recurse from linear equality $start$ with depth $depth$; $sum$ is the number of trails that have been removed by the $i$th inequality; $depthmin$ is the number of linear inequalities in $path$ */
  function DFS($depth$, $start$, $sum$, $depthmin$)
      if $sum = trailsize$ and $depthmin > depth$ then
          $depthmin \leftarrow depth$
          return
      end if
      for $i = start$ to $len$ and $depth + 1 < depthmin$ do
          $cnt[depth][i] \leftarrow 0$
          for all $trail \in Bset[depth]$ do
              if $v(Coefset[i], trail) \geq 0$ then      ▷ $trail$ can be removed by inequality $i$
                  add $trail$ to $Bset[depth + 1]$
              else
                  $cnt[depth][i]$++
              end if
          end for
          if $cnt[depth][i] > 0$ then
              add $i$ to $path$
              DFS($depth + 1$, $i + 1$, $sum + cnt[depth][i]$, $depthmin$)
          end if
      end for
  end function
```

of S-Box as *len*, and for PRESENT S-Box, *len* is 122. Our target here is to find the minimum number of inequalities from *len* linear equalities in H-Represent to describe the DDT of a S-Box, i.e. to remove all the impossible division trails in *Bset*. Our algorithm of searching is in Algorithm 1. The 8 linear inequalities of PRESENT S-Box we found by our algorithm are:

$$\begin{cases} -a_2 - a_1 + b_3 + b_1 + b_0 \geq -1 \\ -3a_3 - 3a_2 - 3a_1 + b_3 + 2b_2 + b_1 + 2b_0 \geq -5 \\ -2a_3 - a_2 - a_1 - 2a_0 + 5b_3 + 5b_2 + 5b_1 + 2b_0 \geq 0 \\ -a_0 - b_3 - b_2 + 2b_1 - b_0 \geq -2 \\ a_3 + a_2 + a_1 + a_0 - 2b_3 - 2b_2 + b_1 - 2b_0 \geq -1 \\ -a_0 + 2b_3 - b_2 - b_1 - b_0 \geq -2 \\ -a_0 - 2b_3 + b_2 - 2b_1 + b_0 \geq -3 \\ a_3 + a_2 + a_1 + 2a_0 - 2b_2 - 2b_1 - 2b_0 \geq -1 \end{cases}$$

B.3 Model Division Propagation of MixColumns - An example for P_{144}

In this section, we show how to represent the matrix in bit level, giving an example for P_{144}. We remember that given the polynomial of the field where the multiplications operate on, the representation of the matrix is unique. For PHOTON permutations based on $\mathbb{F}_{2^4}$, the polynomial is $x^4 + x + 1$.

The MDS matrix of P_{144} is as follows

$$\mathrm{MC}_{144} = \begin{pmatrix} 0&1&0&0&0&0\\ 0&0&1&0&0&0\\ 0&0&0&1&0&0\\ 0&0&0&0&1&0\\ 0&0&0&0&0&1\\ 1&2&8&5&8&2 \end{pmatrix}^6 = \begin{pmatrix} 1&2&8&5&8&2\\ 2&5&1&2&6&12\\ 12&9&15&8&8&13\\ 13&5&11&3&10&1\\ 1&15&13&14&11&8\\ 8&2&3&3&2&8 \end{pmatrix}$$

can be easily rewritten into $\mathrm{MC}_{144} = (m_{ij})_{24\times 24}$ as

$$\left(\begin{array}{cccc|cccc|cccc|cccc|cccc|cccc}
1&0&0&0&0&1&0&0&1&0&0&1&1&0&1&0&1&0&0&1&0&1&0&0\\
0&1&0&0&0&0&1&0&1&1&0&0&1&1&0&1&1&1&0&0&0&0&1&0\\
0&0&1&0&1&0&0&1&0&1&1&0&1&1&1&0&0&1&1&0&1&0&0&1\\
0&0&0&1&1&0&0&0&0&0&1&0&0&1&0&1&0&0&1&0&1&0&0&0\\
\hline
0&1&0&0&1&0&1&0&1&0&0&0&0&1&0&0&0&1&1&0&1&0&1&1\\
0&0&1&0&1&1&0&1&0&1&0&0&0&0&1&0&1&0&1&1&0&1&0&1\\
1&0&0&1&1&1&1&0&0&0&1&0&1&0&0&1&0&1&0&1&1&0&1&0\\
1&0&0&0&0&1&0&1&0&0&0&1&1&0&0&0&1&1&0&0&0&1&1&0\\
\hline
1&0&1&1&0&0&0&1&0&1&1&1&1&0&0&1&1&0&0&1&0&0&1&1\\
0&1&0&1&1&0&0&0&0&0&1&1&1&1&0&0&1&1&0&0&0&0&0&1\\
1&0&1&0&0&1&0&0&0&0&0&1&0&1&1&0&0&1&1&0&1&0&0&0\\
0&1&1&0&0&0&1&1&1&1&1&1&0&0&1&0&0&0&1&0&0&1&1&1\\
\hline
0&0&1&1&1&0&1&0&0&1&0&1&1&1&0&0&1&1&0&1&1&0&0&0\\
0&0&0&1&1&1&0&1&1&0&1&0&0&1&1&0&1&1&1&0&0&1&0&0\\
1&0&0&0&1&1&1&0&1&1&0&1&1&0&1&1&1&1&1&1&0&0&1&0\\
0&1&1&1&0&1&0&1&1&0&1&1&1&0&0&1&1&0&1&0&0&0&0&1\\
\hline
1&0&0&0&0&1&1&1&0&0&1&1&1&1&1&1&0&1&0&1&1&0&0&1\\
0&1&0&0&0&0&1&1&0&0&0&1&0&1&1&1&1&0&1&0&1&1&0&0\\
0&0&1&0&0&0&0&1&1&0&0&0&0&0&1&1&1&1&0&1&0&1&1&0\\
0&0&0&1&1&1&1&1&0&1&1&1&1&1&1&0&1&0&1&1&0&0&1&0\\
\hline
1&0&0&1&0&1&0&0&1&1&0&0&1&1&0&0&0&1&0&0&1&0&0&1\\
1&1&0&0&0&0&1&0&0&1&1&0&0&1&1&0&0&0&1&0&1&1&0&0\\
0&1&1&0&1&0&0&1&1&0&1&1&1&0&1&1&1&0&0&1&0&1&1&0\\
0&0&1&0&1&0&0&0&1&0&0&1&1&0&0&1&1&0&0&0&0&0&1&0
\end{array}\right)$$

C Zero-sums of P_{100} for PHOTON-80/20/16

C.1 10-round Zero-sums for P_{100} of Size 2^{40}

By tool, we found the following:

$$\mathbb{PB} \xleftarrow{R^{-4}} \begin{pmatrix} A & C\,C\,C\,C\\ A & C\,C\,C\,C\\ A & C\,C\,C\,C\\ A & C\,C\,C\,C\\ aaac & C\,C\,C\,C \end{pmatrix},\ \mathbb{B} \xleftarrow{R^{-4}} \begin{pmatrix} A&C&C&C&C\\ A&C&C&C&C\\ A&C&C&C&C\\ A&C&C&C&C\\ A&C&C&C&C \end{pmatrix},\ \begin{pmatrix} A&C&C&C&C\\ C&A&C&C&C\\ C&C&A&C&C\\ C&C&C&A&C\\ C&C&C&C&A \end{pmatrix} \xrightarrow{R^5} \mathbb{B}.$$

By adding one more round in the middle, one can construct 10-round zero-sums for P_{100} of size 2^{40} , while it is possible to construct a 9-round balance zero-sum for P_{100} with size 2^{36} and a 9-round partial balance zero-sum for P_{100} with partition size of 2^{35}.

C.2 11-round Zero-sums of P_{100} with 2^{76} Texts

By tool, we found the following:

$$\mathbb{B} \xleftarrow{R^{-4}} \begin{pmatrix} A & C & C & C & C \\ A & C & C & C & C \\ A & C & C & C & C \\ A & C & C & C & C \\ A & C & C & C & C \end{pmatrix}, \begin{pmatrix} A & C & A & A & A \\ A & A & C & C & A \\ A & A & A & C & C \\ A & A & A & A & C \\ C & A & A & A & A \end{pmatrix} \xrightarrow{R^{6}} \mathbb{B}.$$

By adding one more round in the middle, one can construct a 11-round zero-sums for P_{100} with 2^{76} texts.

D Zero-sums of P_{196} for PHOTON-160/36/36

D.1 10-round zero-sum partition of P_{196} with 2^{48} Texts

With the tool, we found the following:

$$\mathbb{PB} \xleftarrow{R^{-4}} \begin{pmatrix} A & C & C & C & C & C & C \\ A & C & C & C & C & C & C \\ A & C & C & C & C & C & C \\ A & C & C & C & C & C & C \\ A & C & C & C & C & C & C \\ aaac & C & C & C & C & C & C \\ C & C & C & C & C & C & C \end{pmatrix}, \begin{pmatrix} A & C & C & C & C & C & C \\ C & A & C & C & C & C & C \\ C & C & A & C & C & C & C \\ C & C & C & A & C & C & C \\ C & C & C & C & A & C & C \\ C & C & C & C & C & A & C \\ C & C & C & C & C & C & C \end{pmatrix} \xrightarrow{R^{5}} \mathbb{B}$$

and

$$\mathbb{B} \xleftarrow{R^{-4}} \begin{pmatrix} A & C & C & C & C & C & C \\ A & C & C & C & C & C & C \\ A & C & C & C & C & C & C \\ A & C & C & C & C & C & C \\ A & C & C & C & C & C & C \\ A & C & C & C & C & C & C \\ C & C & C & C & C & C & C \end{pmatrix}, \begin{pmatrix} A & C & C & C & C & C & C \\ C & A & C & C & C & C & C \\ C & C & A & C & C & C & C \\ C & C & C & A & C & C & C \\ C & C & C & C & A & C & C \\ C & C & C & C & C & A & C \\ C & C & C & C & C & C & C \end{pmatrix} \xrightarrow{R^{5}} \mathbb{B}.$$

By adding one more round in the middle, one can construct a 10-round zero-sum partition for P_{196} with 2^{48} texts, while it is possible to construct a 9-round partial zero-sum partition for P_{196} with 2^{43} texts and a balance zero-sum partition for P_{196} with size of 2^{44} texts.

D.2 11-round zero-sum partition of P_{196} with 2^{108} Texts

By tool, we found the following:

$$\mathbb{B} \xleftarrow{R^{-4}} \begin{pmatrix} A & C & C & C & C & C & C \\ A & C & C & C & C & C & C \\ A & C & C & C & C & C & C \\ A & C & C & C & C & C & C \\ A & C & C & C & C & C & C \\ A & C & C & C & C & C & C \\ C & C & C & C & C & C & C \end{pmatrix}, \begin{pmatrix} A & C & C & C & C & A & A \\ A & A & C & C & C & C & A \\ A & A & A & C & C & C & C \\ C & A & A & A & C & C & C \\ C & C & A & A & A & C & C \\ C & C & C & A & A & A & C \\ C & C & C & C & A & A & A \end{pmatrix} \xrightarrow{R^{6}} \mathbb{B}.$$

By adding one more round in the middle, one can construct a 11-round zero-sum partition for P_{196} with 2^{108} texts.

D.3 12-round (Partial) Zero-sum Partitions for P_{196} of Size 2^{159}

By tool, we found that it is only possible to set up a Partial Balance 12-round distinguisher of P_{196}, with a complexity of 2^{159} texts:

$$\mathbb{PB} \xleftarrow{R^{-6}} \begin{pmatrix} A & A & A & A & A & C & C \\ A & A & A & A & A & C & C \\ A & A & A & A & A & aaac & C \\ A & A & A & A & A & A & C \\ A & A & A & A & A & A & C \\ A & A & A & A & A & A & C \\ A & A & A & A & A & A & C \end{pmatrix} \xrightarrow{R^{6}} \mathbb{B}.$$

E Zero-sums of P_{256} for PHOTON-224/32/32

We remember that the 12-round distinguisher is presented in details in Sect. 5.4.

E.1 10-round Zero-sum partition for P_{256} of Size 2^{63}

By tool, we found the following:

$$\mathbb{B} \xleftarrow{R^{-4}} \begin{pmatrix} A & C & C & C & C & C & C & C \\ A & C & C & C & C & C & C & C \\ A & C & C & C & C & C & C & C \\ A & C & C & C & C & C & C & C \\ A & C & C & C & C & C & C & C \\ A & C & C & C & C & C & C & C \\ aaac & C & C & C & C & C & C & C \\ C & C & C & C & C & C & C & C \end{pmatrix}, \begin{pmatrix} A & C & C & C & C & C & C & C \\ C & A & C & C & C & C & C & C \\ C & C & A & C & C & C & C & C \\ C & C & C & A & C & C & C & C \\ C & C & C & C & A & C & C & C \\ C & C & C & C & C & A & C & C \\ C & C & C & C & C & C & aaac & C \\ C & C & C & C & C & C & C & C \end{pmatrix} \xrightarrow{R^{5}} \mathbb{B}.$$

By adding one more round in the middle, one can construct a 10-round zero-sum partition for P_{256} with 2^{54} texts, while it is possible to construct a 9-round zero-sum partition for P_{256} with 2^{50} texts.

E.2 11-round Zero-sum Partition for P_{256} of Size 2^{119}

By tool, we found the following:

$$\mathbb{B} \xleftarrow{R^{-4}} \begin{pmatrix} A & C & C & C & C & C & C & C \\ A & C & C & C & C & C & C & C \\ A & C & C & C & C & C & C & C \\ A & C & C & C & C & C & C & C \\ A & C & C & C & C & C & C & C \\ A & C & C & C & C & C & C & C \\ A & C & C & C & C & C & C & C \\ aaac & C & C & C & C & C & C & C \end{pmatrix}, \begin{pmatrix} A & C & C & C & C & C & A & A \\ A & A & C & C & C & C & C & C \\ A & A & A & C & C & C & C & C \\ C & A & A & A & C & C & C & C \\ C & C & A & A & A & C & C & C \\ C & C & C & A & A & A & C & C \\ C & C & C & C & A & A & A & C \\ C & C & C & C & C & A & A & A \end{pmatrix} \xrightarrow{R^{6}} \mathbb{B}.$$

By adding one more round in the middle, one can construct a 11-round zero-sum partition for P_{256} with 2^{119} texts.