Algebraic Cryptanalysis of Frit

Christoph Dobraunig[1], Maria Eichlseder[1], Florian Mendel[2] and Markus Schofnegger[1]

[1] Graz University of Technology, Austria
[2] Infineon Technologies AG, Germany
maria.eichlseder@iaik.tugraz.at

Abstract. Frit is a cryptographic 384-bit permutation recently proposed by Simon et al. and follows a novel design approach for built-in countermeasures against fault attacks. We analyze the cryptanalytic security of Frit in different use-cases and propose attacks on the full-round primitive. We show that the inverse Frit^{-1} of Frit is significantly weaker than Frit from an algebraic perspective, despite the better diffusion of the inverse of the used mixing functions σ: Its round function has an effective algebraic degree of only about 1.325. We show how to craft structured input spaces to linearize up to 4 (or, conditionally, 5) rounds and thus further reduce the degree. As a result, we propose very low-dimensional start-in-the-middle zero-sum partitioning distinguishers for unkeyed Frit, as well as integral distinguishers for round-reduced Frit and full-round Frit^{-1}. We also consider keyed Frit variants using Even-Mansour or arbitrary round keys. By using optimized interpolation attacks and symbolically evaluating up to 5 rounds of Frit^{-1}, we obtain key-recovery attacks with a complexity of either 2^{59} chosen plaintexts and 2^{67} time, or 2^{18} chosen ciphertexts and time (about 5 seconds in practice).

Keywords: Cryptanalysis · Frit · higher-order differentials · interpolation attack

1 Introduction

Attacks that target the implementation of a scheme, such as side-channel [Koc96, KJJ99] and fault attacks [BDL97, BS97], are a threat to cryptographic security in practice, especially in situations where an attacker has physical access to the device performing the cryptographic computations. In order to mitigate such attacks, a variety of countermeasures has been proposed, such as masking [CJRR99, GP99] and threshold implementations [NRR06, NRS08, NRS11] to protect against side-channel attacks, or the integration of some form of error detection [SMG16] to protect against fault attacks. The overhead cost of implementing these countermeasures typically depends on properties of the cryptographic primitive, such as its multiplicative complexity in the case of masking. This has motivated cryptographers to design primitives that minimize these costs. For example, Noekeon [DPVR00], Keccak [BDPV11], or Ascon [DEMS14] aim to reduce the cost of masking countermeasures by using low degree S-boxes, while other designs like Zorro [GGNPS13] even use incomplete S-box layers in order to be easier to mask, i.e., only part of the state is updated by the S-box layer, the rest remains unchanged.

The recently proposed permutation Frit [SBD+18] takes this approach further and does not only allow efficient masking, but has been designed to also provide low-cost built-in fault detection. Frit is a 384-bit cryptographic permutation designed by Simon, Batina, Daemen, Grosso, Massolino, Papagiannopoulos, Regazzoni, and Samwel [SBD+18]. The round function uses 128 AND-gates per round as its only source of non-linearity. Its operations are carefully chosen to minimize the cost of maintaining an additional

128-bit checksum of the current state to provide redundancy and detect faults. As a result, even protected implementations with both side-channel and fault countermeasures are still relatively lightweight. With its 384-bit blocksize, it is well-suited as a building block for the modes of permutation-based cryptography, such as sponge and duplex modes [BDPV07, BDPV08, BDPV12], but it can also be transformed into a big-state Even-Mansour block cipher [EM91].

Related Work. As a consequence of the design choices, Frit shares some similarities with constructions like Zorro and LowMC [ARS+15] that have incomplete S-box layers. Clearly, such novel designs require third-party cryptanalysis in order to strengthen the trust in their security, or to learn how to improve for future designs. Zorro paved the way for interesting cryptanalytic results that exploit the existence of good differential or linear characteristics in such incomplete S-box layers [RASA14, WWGY14, LMR15], as well as invariant subspace attacks [LMR15]. On the other hand, the analysis results [DEM15, DLMW15] for LowMC exploit the low degree of its round function together with its partial S-box layer.

The increasing prominence of designs with low-degree round functions, such as Keccak [BDPV11], Ketje [BDP+16], Keyak [BDP+14], Ascon [DEMS14], Xoodoo [DHVV18], or Gimli [BKL+17], as well as more experimental designs that aggressively minimize the number of AND-gates, such as Flip [MJSC16], Kreyvium [CCF+16, CCF+18], LowMC [ARS+15], or Rasta [DEG+18], has led to many advances and insights in algebraic cryptanalysis. Examples include extensions of cube attacks [DS09], such as cube-like attacks [DMP+15, DLWQ17] and conditional cube attacks [LDW17, HWX+17, LBDW17], but also many variants that exploit the algebraic properties in other ways, like collision attacks [SLG17] and preimage attacks [GLS16] on round-reduced Keccak that linearize parts of its underlying permutation. Moreover, new techniques like the division property [Tod15b, Tod15a] as a generalization of the integral attack [KW02] have been recently proposed to construct integral distinguishers further exploiting low-degree round functions.

Contributions. We analyze the security of Frit and provide distinguishers for the unkeyed primitive as well as key-recovery attacks for keyed Frit. Our analysis takes advantage of the relatively low algebraic degree of Frit's round function, but even more so of the properties of its inverse Frit^{-1}, including its algebraic degree and certain diffusion properties. As observed by the designers of Frit, the algebraic degree of the Frit round function is 2, but an upper bound on the algebraic degree of multi-round Frit is given by the Fibonacci sequence. It can thus be argued that the effective degree of its round function, i.e., the growth rate of the degree over multiple rounds, is the golden ratio $\varphi \approx 1.618$, and at least 11 (out of 16) rounds of Frit are necessary to reach a degree larger than 128, while 13 rounds are necessary to reach the maximum degree of 383. The same upper bound can be shown for Frit^{-1}. However, we show that this bound is far from tight, and prove an upper bound corresponding to an effective degree of $\alpha_0 \approx 1.325$. This observation implies that the algebraic degree of 16-round Frit^{-1} is only 114. We show how to craft initial structures that linearize up to 4 rounds of Frit^{-1} (or 5 rounds under certain additional bit-conditions on the input).

Furthermore, we analyze the use of Frit as an Even-Mansour [EM91] block cipher. If we allow chosen-ciphertext queries, we can take advantage of the properties of Frit^{-1} to recover the key using 2^{18} chosen ciphertexts in about 5 seconds. However, since Frit^{-1} is generally more costly to evaluate than Frit, it seems more likely that Frit would in practice be used in a construction that allows only chosen-plaintext queries. For this potential use case, we propose an optimized interpolation attack [DLMW15]. We take advantage of the relatively low algebraic degree of Frit to set up an integral distinguisher for 11 rounds and combine this with a 5-round key recovery technique using interpolation. The complexity of the interpolation profits not only from the very low degree of Frit^{-1}, but also from its limited diffusion that leads to a rather low monomial count when expressing intermediate state bits as a function of the ciphertext and key bits. With this approach, we can recover the Even-Mansour key for full-round Frit using 2^{59} chosen plaintexts and about 2^{67} time.

In Section 2, we briefly describe the Frit design. In Section 3, we analyze the algebraic degree of Frit and Frit^{-1} and propose initial structures to linearize several rounds. Based on these properties, we propose key-recovery attacks on keyed Frit in Section 4.

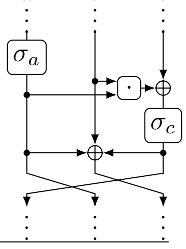
2 Description of Frit

Frit is a cryptographic permutation designed by Simon et al. [SBD+18]. Its 384-bit state is divided into three 128-bit limbs a, b, c which are updated in 16 rounds using simple bitwise operations, as illustrated in Figure 1 (left). The only nonlinear operation is one 128-bit bitwise AND ($\odot$) per round, used in a Toffoli gate. Diffusion is achieved by two rotation-invariant linear mixing functions using 128-bit bitwise XOR ($\oplus$) and bitwise circular left shifts ($\lll$), which we refer to as σ_a and σ_c. Both functions σ_a, σ_c compute each output bit as the XOR of 3 input bits and have a bitwise branch number of 4 bits. The 16 rounds are identical except for the value of the round constant RC_r.

Frit (for "Fault-Resistant Iterative Transformation") was designed to support the implementation of countermeasures against physical attacks. The design follows a more general approach proposed by its designers to provide built-in protection against differential fault attacks (DFA). The core idea of this approach is to extend the state by an extra limb and to implement an extended round function that updates all limbs such that the XOR of all limbs remains constant. The operations in the Frit round function were selected such that this extended round function is very efficient, and they are additionally well-suited for side-channel countermeasures such as threshold implementations (TI). Our attack is however independent of implementation details such as the extra limb, so we refer to the original design paper for the detailed specification [SBD+18].

In Figure 1 (right), we also list Frit's inverse, Frit^{-1}. Inverting corresponds to executing the operations in reverse order, where the Feistel swap is reversed and the mixing functions σ_a, σ_c are replaced with their inverses $\sigma_a^{-1}, \sigma_c^{-1}$. These inverses are again rotation-invariant, but they require significantly more operations: while σ_a, σ_c XOR 3 rotated copies of the input, σ_a^{-1} requires 65 rotations and σ_c^{-1} requires 33 (since σ_c has only even rotation constants and can thus be partitioned into the parallel application of two 64-bit σ functions). Frit is thus more likely to be used in modes and constructions that do not require the inverse. Its 384-bit size seems well-suited for sponge and duplex modes [BDPV07, BDPV08, BDPV12], but it can also be transformed into a big-state Even-Mansour block cipher [EM91].

```
Frit(a, b, c)
Input: $a, b, c \in \{0,1\}^{128}$
for $r = 0, \dots, 15$ do
    $a \leftarrow a \oplus (a \lll 110) \oplus (a \lll 87)$
    $c \leftarrow c \oplus (a \odot b) \oplus \text{RC}_r$
    $c \leftarrow c \oplus (c \lll 118) \oplus (a \lll 88)$
    $b \leftarrow a \oplus b \oplus c$
    $(a, b, c) \leftarrow (c, a, b)$
return $(a, b, c)$
```



```
Frit^{-1}(a, b, c)
Input: $a, b, c \in \{0,1\}^{128}$
for $r = 15, \dots, 0$ do
    $(c, a, b) \leftarrow (a, b, c)$
    $b \leftarrow a \oplus b \oplus c$
    $c \leftarrow \sigma_c^{-1}(c)$
    $c \leftarrow c \oplus (a \odot b) \oplus \text{RC}_r$
    $a \leftarrow \sigma_a^{-1}(a)$
return $(a, b, c)$
```

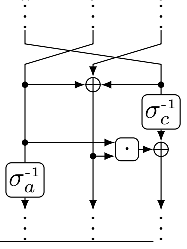


Figure 1: The permutation $\text{Frit}(a, b, c)$ [SBD+18] and its inverse $\text{Frit}^{-1}(a, b, c)$.

3 Algebraic Degree of Frit and Frit^{-1}

In this section, we analyze the algebraic degree of r-round Frit and Frit^{-1}. We show that the degree of Frit^{-1} grows significantly more slowly than that of Frit. We introduce the notion of the "effective degree" of the round function as the growth rate of the degree. We show that the effective degree of Frit is bounded by $\varphi \approx 1.618$ and of Frit^{-1} by $\alpha_0 \approx 1.325$, while the generic bound for degree-2 round functions is 2.

3.1 Designers' Analysis of Frit

In an appendix of the Frit paper [SBD+18], the designers analyze the algebraic degree of r-round Frit, which we denote by Frit$_r$, and observe the following. Let

$$(a_r, b_r, c_r) = \textsc{Frit}_1(a_{r-1}, b_{r-1}, c_{r-1}) = \textsc{Frit}_r(a_0, b_0, c_0)\,.$$

Let $F_0 = 0, F_1 = 1$, and $F_i = F_{i-1} + F_{i-2}$ for $i \geq 2$ denote the Fibonacci sequence. Then, using the definition of Frit$_1$ and the initial conditions $\deg a_0 = \deg b_0 = \deg c_0 = 1$, it is easy to see by induction that $\deg a_r \leq F_{r+2}$, $\deg c_r \leq F_{r+2}$, and $\deg b_r \leq F_{r+1}$:

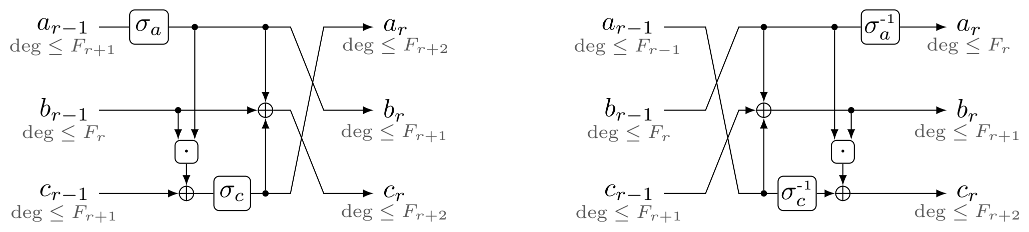
$$\begin{aligned} \deg b_r &= \deg \sigma_a(a_{r-1}) \leq F_{r+1}\,, \\ \deg a_r &= \deg \sigma_c(c_{r-1} \oplus b_{r-1} \odot \sigma_a(a_{r-1})) \leq F_r + F_{r+1} = F_{r+2}\,, \\ \deg c_r &= \deg(b_{r-1} \oplus b_r \oplus a_r) \leq F_{r+2}\,. \end{aligned} \tag{Frit}$$

A similar bound applies for Frit$_r^{-1}$, where we obtain with the same reasoning that

$$\deg c_r \leq F_{r+2}\,, \qquad \deg b_r \leq F_{r+1}\,, \qquad \deg a_r \leq F_r\,, \tag{Frit^{-1}}$$

except for the initial condition $\deg a_0 = 1$. Thus, $d_r = F_{r+2}$ is an upper bound for the algebraic degree $\deg \textsc{Frit}_r \leq d_r$ and $\deg \textsc{Frit}_r^{-1} \leq d_r$.

Since $F_{15} = 610 \geq 383$, at least 13-round (14-round) Frit or 13-round (15-round) Frit^{-1} is necessary to achieve the maximum degree in some (all) limbs of the state.



(a) Round r of Frit (b) Round r of Frit^{-1}

Figure 2: Fibonacci bound on the degree of Frit$_r$ [SBD+18] and Frit$_r^{-1}$.

3.2 Algebraic Degree of Frit^{-1}

In the following, we have a closer look at Frit^{-1} to derive a tighter recursive bound on the degree of Frit^{-1} that looks quite similar to the case of Frit and shares the same initial conditions, but grows significantly more slowly.

3.2.1 Recursive Bound for the Degree

To see that the previous bound is not tight for Frit^{-1}, consider the first two rounds of the inverse. Note that σ and its inverses do not change the degree, so we write $\bar{x}$ for any

$\sigma(x)$ or $\sigma^{-1}(x)$. After one round, the algebraic degrees of limbs (a_1, b_1, c_1) are $(1, 1, 2)$ since $c_1 = b_1 \odot \bar{a}_1 \oplus \bar{a}_0$. Now consider c_2 after two rounds, which is essentially computed as $c_2 = b_1 \odot (c_1 \oplus \dots) \oplus \dots$ and thus has a degree of at most $1 + 2 = 3$. However, c_1 is itself the result of a multiplication by b_1, and since $b_1^2 = b_1$, the actual degree of the result is only 2, not 3.

More generally, using the bound d_r defined by the recursion $d_r = d_{r-2} + d_{r-3}$ and the initial conditions $d_0 = d_{-1} = d_{-2} = 1$, the degree of $\textsc{Frit}^{-1}$ is bounded by

$$\deg c_r \le d_r , \qquad \deg b_r \le d_{r-1} , \qquad \deg a_r \le d_{r-2} .$$

We can prove this inductively by using the fact that by definition, $c_r = b_r \odot \bar{a}_r \oplus \bar{a}_{r-1}$:

$$\begin{aligned} \deg a_r &= \deg \bar{b}_{r-1} \le d_{r-2} , \\ \deg b_r &= \deg (c_{r-1} \oplus b_{r-1} \oplus a_{r-1}) \le d_{r-1} , \\ \deg c_r &= \deg (\bar{a}_{r-1} \oplus b_{r-1} \odot (a_{r-1} \oplus b_{r-1} \oplus c_{r-1})) \\ &= \deg (\bar{a}_{r-1} \oplus b_{r-1} \odot (a_{r-1} \oplus 1 \oplus \bar{a}_{r-1} \oplus \bar{a}_{r-2})) \le d_{r-2} + d_{r-3} = d_r . \end{aligned} \tag{$\textsc{Frit}^{-1}$}$$

In summary, we obtain the recursion

$$\deg \textsc{Frit}_r^{-1} \le d_r = d_{r-2} + d_{r-3} , \qquad d_0 = d_{-1} = d_{-2} = 1 .$$

Using the method of differences to rewrite $d_r = d_{r-1} + (d_r - d_{r-1})$, we can also derive a different recursion for $\textsc{Frit}^{-1}$ very similar to that of $\textsc{Frit}$:

$$\begin{aligned} \deg \textsc{Frit}_r &\le d_r = d_{r-1} + d_{r-2} , & d_1 &= 2, \quad d_0 = 1 , \\ \deg \textsc{Frit}_r^{-1} &\le d_r = d_{r-1} + d_{r-2} - d_{r-4} , & d_1 &= 2, \quad d_0 = d_{-1} = d_{-2} = 1 . \end{aligned}$$

Despite the apparent similarity of the recursions, the tighter bound for $\textsc{Frit}^{-1}$ grows significantly more slowly, as we will discuss in the following (see Figure 3). We practically verified the resulting degrees for up to 4 rounds of $\textsc{Frit}^{-1}$ by symbolically evaluating the cipher with `Sage`, and the bound of degree 4 is tight. We also verified up to 7 rounds (degree 9) symbolically with a simplified model of the cipher, as well as up to 11 rounds (degree 28) by testing the zero-sum property, and all results confirm these bounds.

3.2.2 Effective Degree

The recursive definition of the bound identified above can also be translated to a closed-form expression, in analogy to the bound of $d_r = 2^r$ for the degree after r rounds of degree $d = 2$. For the permutation $\textsc{Frit}$, the designers' Fibonacci argument that we recalled in Subsection 3.1 yields the following explicit exponential form using Binet's formula:

$$d_r = F_{r+2} = \frac{\varphi^{r+2} - (1-\varphi)^{r+2}}{\sqrt{5}} = \left\lfloor \frac{\varphi^{r+2}}{\sqrt{5}} \right\rceil ,$$

where $\varphi = \frac{1+\sqrt{5}}{2}$ is the golden ratio and $\lfloor \cdot \rceil$ denotes rounding to the nearest integer. We thus refer to $\varphi \approx 1.618$ as (an upper bound d for) the effective degree of $\textsc{Frit}$.

We can obtain a similar exponential form for $\textsc{Frit}^{-1}$ by considering the generating function $D(z) \in \mathbb{C}[z]$ of the recursive form $d_r = d_{r-2} + d_{r-3}$:

$$D(z) = z^3 - z - 1.$$

The polynomial $D(z)$ has three roots $\alpha_0, \alpha_1, \alpha_2$ over the complex plane. Two roots α_1, α_2 are complex with absolute value less than 1, only one root $\alpha_0 \approx 1.325$ is real. It is well-known that d_r can be written as a linear combination of powers of these roots, where

		$r = 0$	1	2	3	4	5	6	7	8	9	…
$d = 2$	$d_r = 2d_{r-1}$	$= 1$	2	4	8	16	32	64	128	256	512	…
$d = \varphi \approx 1.618$	$d_r = d_{r-1} + d_{r-2}$	$= 1$	2	3	5	8	13	21	34	55	89	…
$d = \alpha_0 \approx 1.325$	$d_r = d_{r-2} + d_{r-3}$	$= 1$	2	2	3	4	5	7	9	12	16	…

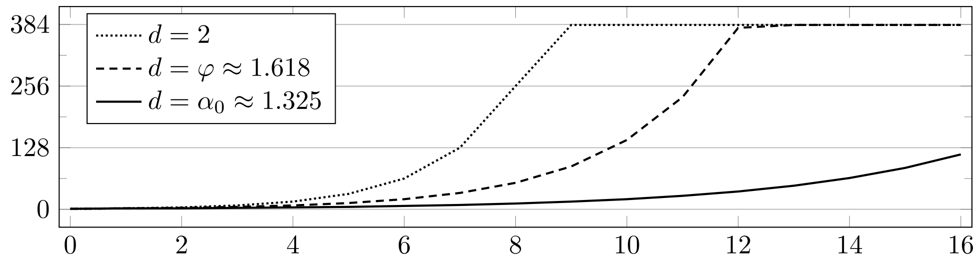


Figure 3: Upper bounds d_r on $\deg \textsc{Frit}_r^{-1}$ based on different effective degrees d.

the coefficients t_0, t_1, t_2 depend on the initial conditions d_0, d_1, d_2. By solving the resulting system of three linear equations, we obtain that $|t_1\alpha_1^r + t_2\alpha_2^r| < 0.4$ for all $r \geq 0$ and thus the effective degree is (bounded by) $d = \alpha_0$:

$$d_r = t_0\alpha_0^r + t_1\alpha_1^r + t_2\alpha_2^r = \lfloor t_0\alpha_0^r \rceil\,, \qquad t_0 \approx 1.267, \quad \alpha_0 \approx 1.325.$$

Figure 3 compares the resulting degrees after r rounds for effective degrees 2, φ, and α_0.

3.3 (Conditional) Initial Structures and Integral Distinguishers

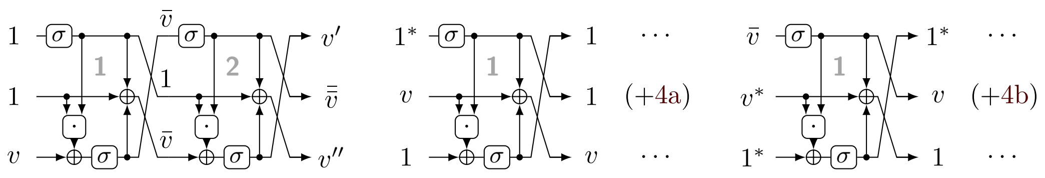
So far, we analyzed the algebraic degree of $\textsc{Frit}_r$ and $\textsc{Frit}_r^{-1}$ with respect to the 384 input variables representing the permutation input (or output). While the bound for the degree of any of the limbs is below the generic bound of 383 that holds for any bijective operation, we can use this property to distinguish the permutation. More specifically, we can use the methods of higher-order differential cryptanalysis [Lai94] to obtain an integral distinguisher: Say we have an upper bound d on the degree of the r-round permutation $\textsc{Frit}_r$ or $\textsc{Frit}_r^{-1}$ (or on at least one limb of the state). If we apply the r-round permutation to all elements of some $(d+1)$-dimensional (affine) subspace of $\mathbb{F}_2^{384}$ and compute the XOR of the resulting outputs, we will obtain 0 in all bit positions of the state (or limb).

In the following, we will extend these simple integral distinguishers by several rounds. We will craft structured $(d+1)$-dimensional affine input subspaces such that applying s rounds of the permutation will again produce a $(d+1)$-dimensional affine subspace as an intermediate result, thus extending the integral distinguisher to $s + r$ rounds. In other words, if we write the structured input space V as a linear combination $V = \{\sum v_i \cdot b_i\} = \{B \cdot v \mid v \in \mathbb{F}_2^{d+1}\}$ of some basis vectors b_i, $0 \leq i < d+1$, then the s-round permutation is a linear function with respect to the coordinates v_i. We thus consider the ANF after s rounds when substituting the appropriate linear combination of v_i plus a symbolic constant for each input variable, and show that the resulting degree with respect to the variables v_i is 1. We also consider conditional initial structures where we require b bit conditions on the constants (the key) to ensure that the s-round permutation is linear. Finally, we propose low-dimensional inside-out zero-sum partitioning distinguishers for the permutation.

In the remainder of the section, we use the following notation. For simplicity, we will liberally refer to tuples of elements as "vectors" and to modules, affine vectorspaces, etc. as "spaces". We denote bitwise XOR by $\oplus$, bitwise AND by $\odot$, and the number of non-zero coordinates of a vector by $\mathrm{wt}(\cdot)$. We consider the 384-bit state and each 128-bit limb as a vector of polynomials in the variables v_i, $0 \leq i < d+1$, i.e., as an element of the 384-dimensional or 128-dimensional space over $\mathbb{F}_2[v_0, \ldots, v_d]$. However, as we will later consider keyed Frit variants in an Even-Mansour construction, the coefficients may depend on a constant 384-bit key $K = (k_0, \ldots, k_{383})$ and thus be unknown to the attacker, so we sometimes variably use the base ring $(\mathbb{F}_2[k_0, \ldots, k_{383}])[v_0, \ldots, v_d]$. We refer to the graded part of degree j $(0 \le j \le d+1)$ of a polynomial with respect to the variables v_i as $\mathrm{d}_j(\cdot)$; for example, $\mathrm{d}_1(k_0 + k_1v_1 + v_2 + v_1v_2) = k_1v_1 + v_2$. We want to find s-round initial structures such that $\mathrm{d}_j(a_s, b_s, c_s) = 0$ for $j \ge 2$. We will identify the polynomial vector x with $\mathrm{d}_j(x) = 0, j \ge 2$ with the affine vector space $\mathrm{d}_0(x) + V$ spanned by $d+1$ basis vectors, where the i-th basis vector is obtained by substituting $v_i = 1$, $v_{i'} = 0$ for $i' \ne i$ in $\mathrm{d}_1(x)$.

3.3.1 Initial Structures for Frit

First consider s-round Frit and assume we target some relatively small dimension $d+1 \le 128$, i.e., $r \le 10$. By keeping the two limbs a_0, b_0 constant and limiting the variables v_i to limb c_0, we can easily linearize $s = 2$ rounds of Frit: After one round, $b_1 = \sigma_a(a_0)$ will be constant, while a_1 and c_1 will depend linearly on the variables v_i in c_0. Consequently, in the second round, again no variables are multiplied by the AND-gate, so a_2, b_2, c_2 all depend linearly on the v_i. The structure is illustrated in Figure 4a, where 1 denotes a constant limb and $v, \bar{v}, v', \ldots$ denotes different linear limbs $(\mathrm{d}_{\ge 2}(\cdot) = 0)$. Two limbs x, y denoted by the same symbol share the same linear part $(\mathrm{d}_1(x) = \mathrm{d}_1(y))$, and $\mathrm{d}_1(\bar{x}) = \sigma(\mathrm{d}_1(x))$. In other words, if we apply Frit_2 to all elements of some $(d+1)$-dimensional affine subspace of $\mathbb{F}_2^{384}$ whose basis vectors b_i are all 0 in the first 256 bits, we will obtain another $(d+1)$-dimensional affine subspace as a result after 2 rounds. Note that the basis of this output space depends on the initial constants in a_0 and is thus not necessarily known; in particular, for keyed Frit, the space depends on the key, so the sum over a space of dimension d is no longer a key-independent constant.



(a) 2 rounds (b) 3 rounds, $b = \mathrm{wt}(v)$ (c) 4 rounds, $b = 2\,\mathrm{wt}(v)$

Figure 4: Initial structures to (conditionally) linearize the first $s \le 4$ rounds of Frit.

There are several ways to extend this structure by one or more rounds at the expense of imposing some bit conditions on the constant part of the input. One of them is illustrated in Figures 4b and 4c. To linearize 3 rounds, we prepend one round to the 2-round structure of Figure 4a and start with variables only in b_0, see Figure 4b. Additionally, we require that the constant in a_0, denoted by 1^*, is such that $\sigma_a(a_0)$ is zero in all b bit positions where b_0 is non-constant, $\sigma_a(a_0) \odot \mathrm{d}_1(b_0) = 0$. Then, the output of the AND-gate of the first round is constant, producing exactly the input structure of Figure 4a at the input to the second round. The number of conditions is $b = \mathrm{wt}(\mathrm{d}_1(v)) \ge d+1$. An attacker can either satisfy these conditions directly in an unkeyed setting, or can guess the relevant linear function of the key and repeat the distinguisher 2^b times with suitable plaintexts in a keyed setting.

For a 4-round structure, we can prepend one more round by satisfying a total of $b = 2\,\mathrm{wt}(\mathrm{d}_1(v))$ conditions, as illustrated in Figure 4c: Let $\bar{a}_0 = \sigma_a(a_0)$. First, we require that $\mathrm{d}_1(\bar{a}_0 \odot b_0) = 0$ to ensure that a_1 is constant. Since $\mathrm{d}_1(\bar{a}_0) = \mathrm{d}_1(b_0)$, this can be satisfied with $\mathrm{wt}(\mathrm{d}_1(v))$ bit conditions on $\mathrm{d}_0(b_0)$ using the fact that $(x+0)\cdot(x+1) = x + x = 0$: we simply require $\mathrm{d}_0(b_0) = 1 + \mathrm{d}_0(\bar{a}_0)$ in all relevant bit positions of b_0. Additionally, each of the previous $\mathrm{wt}(\mathrm{d}_1(v))$ conditions from Figure 4b translate to a bit condition that depends nonlinearly on a_0, b_0, but can be satisfied like a linear condition by varying c_0.

Table 1 summarizes the resulting degree after $s + r \leq 16$ rounds of Frit with different initial structures. With unconditional structures, we can distinguish up to 12 rounds (Figure 4a, degree 89) or 13 rounds (degree 377) of Frit. With conditional structures and degree 89, we can distinguish 13 rounds (90 conditions) or 14 rounds (179 conditions).

Table 1: Degree after $r \leq 16$ rounds of Frit using the initial structures of Figure 4.

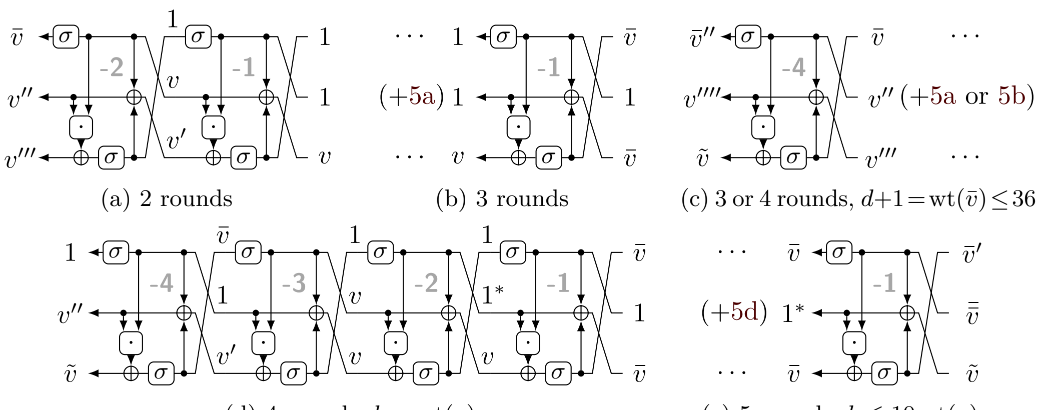
		–	0	1	2	3	4	5	6	7	8	9	10	11	12	**13**	14	15	16
Frit	r	4a	2	3	4	5	6	7	8	9	10	11	**12**	13	14	15	16		
		4b	3	4	5	6	7	8	9	10	11	12	**13**	14	15	16			
		4c	4	5	6	7	8	9	10	11	12	13	**14**	15	16				
	a_r	IS	1	2	3	5	8	13	21	34	55	89	144	233	377	*	*	*	*
	b_r		1	1	2	3	5	8	13	21	34	55	89	144	233	377	*	*	*
	c_r		1	2	3	5	8	13	21	34	55	89	144	233	377	*	*	*	*

3.3.2 Initial Structures for Frit^{-1}

For Frit^{-1}, we can use similar techniques and some additional observations to linearize up to 5 rounds. Figure 5a illustrates an unconditional 2-round structure for Frit^{-1} starting from the same structure as Figure 4a, but with different effects: In the first round, the AND-gate multiplies a linear and a constant limb; the latter acts as a mask such that the non-zero elements of $\mathrm{d}_1(v')$ are a subset of those in $\mathrm{d}_1(v)$. In the second round, the first XOR just inverts this selected subset. Then, in each bit position, the AND-gate either multiplies two linear terms with identical linear parts $\mathrm{d}_1(b_1) = \mathrm{d}_1(b_2)$, or at least one of the inputs is constant. In either case, the result is at most linear, and the non-zero linear part $\mathrm{d}_1(v''')$ is another subset of $\mathrm{d}_1(v)$. We can trivially prepend another round as illustrated in Figure 5b.

Under certain conditions, we can also append a fourth round, as indicated in Figure 5c: Say we are interested in a low dimension of $d + 1 \leq 36$. We start the construction by restricting the linear part $\mathrm{d}_1(\bar{v})$ in limb a_3 (corresponds to a_2 in Figure 5a), and require that it is zero except for the 36 positions $0, \ldots, 17, 64, \ldots, 81$. Now consider $\mathrm{d}_1(c_0) = \mathrm{d}_1(v) = \sigma_a(\mathrm{d}_1(\bar{v}))$: σ_a rotates by $0, 110, 87$ bits to the left, or $0, 18, 41$ bits to the right. Thus, all non-zero elements in $\bar{v}$ are diffused to disjoint bit positions, namely $0 \ldots 17$ to $0 \ldots 17, 18 \ldots 35, 41 \ldots 59$ in the first half of the limb, and $64 \ldots 81$ similarly in the second half. This implies that the linear parts $\mathrm{d}_1(\bar{v}), \mathrm{d}_1(v''), \mathrm{d}_1(v''')$ after 3 rounds are all masked selections from $\mathrm{d}_1(v)$, and the XOR of all three limbs preserves this property. Now, by a similar argument as in Figure 5a, the output of the AND-gate is another selection from $\mathrm{d}_1(v)$ and thus linear. In summary, if we select the linear part at the input as $\mathrm{d}_1(b_0) = 0$ and $\mathrm{d}_1(a_0) = \mathrm{d}_1(c_0) = \sigma_c(\sigma_a(\mathrm{d}_1(\bar{v})))$ with $\mathrm{d}_1(\bar{v})$ zero except in positions $0, \ldots, 17, 64, \ldots, 81$, then we have an initial structure that linearizes 4 rounds of Frit^{-1}.

Alternatively, we can obtain a simpler 4-round structure by imposing bit conditions on the constant part, as illustrated in Figure 5d: For the $\mathrm{wt}(\mathrm{d}_1(v))$ non-zero positions, we require that $\mathrm{d}_0(a_0 \oplus b_0 \oplus c_0) = 1$. Then, we get $\mathrm{d}_1(b_2) = \mathrm{d}_1(c_2)$ and thus $\mathrm{d}_1(b_3) = 0$. We can also prepend another round, at the cost of significantly more new conditions: In addition to the $\mathrm{wt}(\mathrm{d}_1(v))$ conditions on $\mathrm{d}_0(b_2)$ (b_1 in Figure 5d), which we can satisfy by varying a_0 (and compensating in c_0 to keep b_1 constant), we need to fix the behaviour of the AND in the first round. For this purpose, we want to set b_1 to zero in all positions where $\mathrm{d}_1(b_0) = \mathrm{d}_1(\bar{\bar{v}})$ is non-zero, which imposes up to $9\,\mathrm{wt}(\mathrm{d}_1(v))$ additional conditions that can be controlled via c_0. As an example, for $s + r = 16$ we could target $d + 1 = 17$ and would need up to $17 + 92 = 109$ bit conditions for v with 17 consecutive linear bits.



(a) 2 rounds (b) 3 rounds (c) 3 or 4 rounds, $d+1 = \mathrm{wt}(\bar{v}) \leq 36$

(d) 4 rounds, $b = \mathrm{wt}(v)$ (e) 5 rounds, $b \leq 10\,\mathrm{wt}(v)$

Figure 5: Initial structures to (conditionally) linearize the first $s \leq 5$ rounds of Frit^{-1}.

Table 2: Degree after $r \leq 16$ rounds of Frit^{-1} using the initial structures of Figure 5.

		–	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Frit^{-1}	r	5a	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16		
		5b	3	4	5	6	7	8	9	10	11	12	13	14	15	16			
		5c	4	5	6	7	8	9	10	11	12	13	14	15	16				
		5e	5	6	7	8	9	10	11	12	13	14	15	16					
	a_r		1	1	1	2	2	3	4	5	7	9	12	16	21	28	37	49	65
	b_r	IS	1	1	2	2	3	4	5	7	9	12	16	21	28	37	49	65	86
	c_r		1	2	2	3	4	5	7	9	12	16	21	28	37	49	65	86	114

3.3.3 Combined Inside-Out Structure

So far, we considered distinguishing properties of the (possibly keyed) permutation Frit or Frit^{-1}. When considering the permutation as an unkeyed primitive, we can also obtain distinguishing properties that are even less costly to test, but that will usually be less useful to exploit for key-recovery or other attack goals. In particular, we can use inside-out computations and concatenate compatible initial structures to obtain a zero-sum partitioning of very low dimension as follows. The 2-round forward and backward structures of Figure 4a, 5a obviously start from compatible structures $(1, 1, v)$. Furthermore, if we consider a low dimension, we can add a backward round for free as in Figure 5c. As illustrated in Table 3, if we start from $(1, 1, v)$, the degree after 6 forward rounds is at most 8, and after 10 backward rounds at most 9 (for suitable v). Thus, if we fix a suitable v for dimension $d + 1 = 10$, and consider any affine space of this vector space, applying Frit$_{10}^{-1}$ produces $2^{10} = 1024$ inputs to the permutation that sum to 0 while their outputs after Frit$_{16}$ also sum to 0. Since we did not require any bit conditions, we can partition the entire input space of size 2^{384} into 2^{374} such zero-sum partitions of size 2^{10}.

Table 3: Inside-out degrees for 10-dimensional zero-sum partitioning of Frit.

	r	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
In-Out	a_r	5	4	3	2	2	1	1						2	3	5	8
	b_r	7	5	4	3	2	2	1		5c,5a		4a		1	2	3	5
	c_r	9	7	5	4	3	2	2						2	3	5	8

4 Key-Recovery Attacks on Keyed Frit

In this section, we analyze the security of Frit-based block ciphers. In the following attack descriptions, we consider Frit in a single-key Even-Mansour construction to encrypt a 384-bit plaintext P under a 384-bit key K to $C = \text{Frit}(P \oplus K) \oplus K$. The attacks apply with identical complexities for a two-key construction, since the second key can be obtained trivially once the first key has been recovered. For a Frit-like construction with a key schedule and round keys, the attack complexities would also only increase by a negligible amount.

We first propose a simple chosen-ciphertext attack that recovers the key with a very low complexity by exploiting the low effective degree of Frit^{-1} with a 15-round integral distinguisher. However, since the implementation cost of Frit^{-1} is higher than that of Frit, we do not expect that Frit would be used in a way that requires an implementation of the decryption algorithm and thus allows chosen-ciphertext attacks. For this reason, we also propose a chosen-plaintext attack with a higher complexity. While the 11-round integral distinguisher in this case is shorter and weaker due to the higher effective degree of Frit, the key-recovery part can cover more rounds efficiently by applying optimized interpolation attacks that again profit from the low degree of Frit^{-1}.

4.1 Simple Key-Recovery Attack using Chosen Ciphertexts

Assume we can query chosen ciphertexts C to receive the corresponding plaintexts $P = \text{Frit}^{-1}(C \oplus K) \oplus K$. Here and in the following, we always "peel off" the final linear operation σ_a^{-1} by applying σ_a to limb a for all obtained plaintexts, and considering the corresponding equivalent key. If we query 2^{17} ciphertexts in an affine space constructed with the initial structure of Figure 5c, we know that their values of limb a_{15} after decrypting 15 rounds must sum to 0. The relevant degrees after r rounds are summarized in Table 4. Moreover, if we continue decrypting another half-round, the same holds for the value of limb c between the mixing step σ_c^{-1} and the Toffoli gate, which we denote by c^*.

To verify this integral distinguishing property in one bit of c^*, we need to guess 3 bits of (equivalent) key information. Of all 2^3 key guesses, half (including the correct key) will satisfy the 1-bit distinguishing property. After repeating the test 3 or more times for different choices of the initial structure, we expect that only the correct key guess for the 3 bits will survive. The test can be applied in parallel on all 128 bit positions, thus recovering the complete 384-bit key. In order to collect enough data for 3 or more different initial structures, we can query 2^{18} chosen plaintexts and select different 17-dimensional subspaces. For each subspace and bit position, it is sufficient to count how often each of the 2^3 possible values of the 3 relevant plaintext bits occurs; or more specifically, whether it occurs an odd number of times. For each 3-bit key candidate, we can then test whether these $\leq 2^3$ ciphertext values with odd counters sum to the target constant. Overall, the time and data complexity is dominated by cost of querying 2^{18} chosen ciphertexts. An alternative trade-off with a slightly lower data complexity of 2^{14} chosen ciphertexts and a higher, but still practical time complexity can be obtained by using a 14-round distinguisher and guessing 27 key bits.

We practically implemented and verified the attack in C. It takes about 5 seconds to recover the full 384-bit key for the full-round primitive. However, the experiment showed that a minor tweak to the attack is necessary.

Let $K = (K_a, K_b, K_c)$ denote the key we want to recover. Since bit c^* depends linearly on the guessed key bits of K_c, the contribution of these key bits cancels when evaluating the sum, and the distinguishing property is independent of K_c. We can however correctly recover the bits of K_a and K_b one by one with 2 or more repetitions per bit position. To also recover K_c, we tweak the attack as follows. We first recover the full keys K_a, K_b as described above using 2 or more repetitions. Then, we peel off the first round, which is now linear in the remaining unknown key K_c, and repeat the attack for the partially encrypted plaintexts using dimension $12+1$ for 15-round $\textsc{Frit}^{-1}$. This way, we will recover the "equivalent" keys $K'_a = \sigma_a(\sigma_c(K_c))$ and $K'_b = 0$ (though, again, not the part $K'_c = \sigma_c(K_c)$) and thus learn K_c to complete the key K.

Table 4: Key-recovery attacks on keyed $\textsc{Frit}$ (chosen-ciphertext or chosen-plaintext).

		r	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Keyed Frit	CCA	a_r	K	16	12	9	7	5	4	3	2	2	1	1				
		b_r		*	16	12	9	7	5	4	3	2	2	1	5c			
		c_r		*	*	16	12	9	7	5	4	3	2	2				
	CPA	a_r			2	3	5	8	13	21	34	55	*					
		b_r	4a		1	2	3	5	8	13	21	34	55	$f_K(C)$				
		c_r			2	3	5	8	13	21	34	55	*					

4.2 Optimized Interpolation Attack using Chosen Plaintexts

When trying to develop similar key-recovery attacks using chosen plaintexts, we run into several limitations that prevent a full-round attack: First, the degree of $\textsc{Frit}$ grows much faster, so the core distinguisher can only cover fewer rounds. Second, the initial structures we identified in Figure 4 are shorter and only cover two rounds (without additional conditions). Third, the diffusion of σ^{-1} is much better than that of σ, which may impact the size of the necessary key guess. However, we can use a different approach for key-guessing to take advantage of the properties of the weaker inverse $\textsc{Frit}^{-1}$: Jakobsen and Knudsen's interpolation attack [JK97] with Dinur et al.'s variable transformations [DLMW15].

In the following, we propose a chosen-plaintext attack that combines an 11-round integral distinguisher with a 5-round key recovery by interpolation, as illustrated in Table 4. We target the limb $b^* = b_{11}$ after 11 rounds, for which we can construct an integral distinguisher of dimension $55+1$ using an initial structure of $s = 2$ rounds and a core distinguisher for $r = 9$ rounds: the bits of b^* have degree 55 in the selected variables from the plaintext side.

On the other hand, the bits of b^* can also be written as a polynomial of the ciphertext bits with key-dependent coefficients. For 5 rounds, this polynomial can be obtained by considering the ANF of $\textsc{Frit}_5^{-1}$, substituting the keyed ciphertext bits $(K_i \oplus C_i)$ for the $\textsc{Frit}$ output bits, and grouping the terms by the ciphertext monomials in C_i. According to the effective degree of $\textsc{Frit}^{-1}$, this polynomial has degree at most 4 with respect to the variables C_i (see Table 2). We want to recover the key-dependent coefficients of the ciphertext monomials to recover the key. To estimate the complexity of the resulting attack, we first need to analyze the structure and properties of this polynomial for $\textsc{Frit}$.

4.2.1 Interpolation and Monomial Count

We write one bit of b^* as a polynomial $f(K, C)$ of degree 4 in the key $K = (K_0, \dots, K_{383}) \in \mathbb{F}_2^{384}$ and ciphertext $C = (C_0, \dots, C_{383}) \in \mathbb{F}_2^{384}$. This polynomial can be re-written as a key-dependent polynomial in the ciphertext,

$$f_K(C) = \sum \alpha_u C^u \in \mathbb{F}_2[K][C],$$

with ciphertext monomials $C^u = \prod C_i^{u_i}$ for $u = (u_0, \dots, u_{373}) \in \mathbb{F}_2^{384}$ and key-dependent coefficients $\alpha_u \in \mathbb{F}_2[K]$. We are interested in the number of monomials with non-zero coefficients. This number is generally upper-bounded by $\binom{384}{\leq 4} \approx 2^{29.7}$, but is significantly lower for Frit^{-1}. The polynomial is not exactly identical (modulo variable indexing) for different bit positions due to the round constant addition, but the monomial count and structure with respect to the monomials C^u is the same for all bit positions. Also note that due to the Even-Mansour construction, the definition of the polynomial is entirely symmetric with respect to C and K, i.e., $f_K(C) = f_C(K)$.

For the monomial count, we can derive simple bounds by hand, but to obtain a more precise, tight result, we used `Sage` to symbolically evaluate the polynomial. More specifically, we derived the ANF of Frit_r^{-1} for $r \leq 5$, substituted $(K_i \oplus C_i)$ to obtain $f(K, C)$, and counted the number of distinct ciphertext monomials C^u with (potentially) non-zero coefficients. We list the corresponding `Sage` code in Listing 1 in the appendix. For 5 rounds, this takes about 45 minutes, and an upper bound for 6 rounds can be obtained in comparable time. Table 5 lists the resulting exact monomial count $\bar{n}$ for $r \in \{2, 3, 4, 5\}$, grouped by the monomial degree. The polynomial is quite sparse; for Frit_5^{-1}, only $\bar{n} = 60320 \approx 2^{15.88}$ out of the total $\approx 2^{29.7}$ monomials C^u have non-zero coefficients α_u.

We want to interpolate this polynomial and then recover the key bits from its key-dependent coefficients. We could do this with an interpolation attack by collecting $\bar{n}$ equations in the $\bar{n}$ unknown coefficients using $\bar{n}$ different zero-sum distinguishers for our target bit. To improve the complexity, we use the dual approach and variable transformation proposed by Dinur et al. [DLMW15] for the analysis of LowMC. The exact number of monomials of degrees $(0, 1, \ldots, 4)$ is $(1, 337, 30768, 25054, 4160)$. Since the overall degree of $f(K, C)$ is also 4, monomials C^u of high degree in C must have coefficients α_u of low degree in K. We can rewrite the polynomial using fewer unknowns α_u and K^v as

$$f(K, C) = \sum_{\text{wt}(u) \text{ small}} \alpha_u C^u + \sum_{\text{wt}(v) \text{ small}} \beta_v K^v .$$

For Frit_5^{-1}, we can use this approach to reduce the number of unknowns α_u, K^v by a factor of about 2:

- We keep the coefficients of $n_p = 1 + 337 + 30768$ monomials of degree ≤ 2 in C_i.
- We transform the coefficients of the remaining $25054 + 4160$ monomials of degree $3, 4$ in C_i. These coefficients must be polynomials in the key bits of degree at most 1 or 0, respectively. Since the ANF is entirely symmetric in C and K, only 1 or $1 + 337$ key monomials, respectively, can be involved in these coefficients. If we consider Frit with round keys and model them as independent new variables, the count would be higher, but still much smaller than n_p. By linearizing the key monomials, we can replace the $25054 + 4160$ coefficients with $n_k = 1 + 337$ new unknowns.

The reduced number of unknowns is then $n = n_p + n_k = 1 + 2 \cdot 337 + 30768 = 31444 \approx 2^{14.94}$.

Table 5: Number of monomials $\bar{n}$ and unknowns n of $f_K(C)$ per bit for Frit_r^{-1} (Listing 1).

Round	Degree	$\sum$	0	1	2	3	4
2	$\bar{n}$	69	1	67	1		
	n	69	1	68			
3	$\bar{n}$	230	1	163	66		
	n	165	1	164			
4	$\bar{n}$	7921	1	274	7519	127	
	n	550	1	274	275		
5	$\bar{n}$	60320	1	337	30768	25054	4160
	$\log_2(\bar{n})$	15.88	0.00	8.40	14.91	14.61	12.02
	n	31444	1	337	30768	338	
	$\log_2(n)$	14.94	0.00	8.40	14.91	8.40	

Conveniently, the value of 337 of the new unknowns corresponds exactly to different key bits that can thus be trivially recovered from the interpolated polynomial. To recover the full key, it will be necessary to repeat the attack one more time with a different target bit position, which can be done with the same data.

4.3 Key Recovery and Attack Complexity

We state the memory complexity M in bits (bit), data complexity D in the number of queries, and time complexity T in bit operations (op) or encryptions (enc), where $1\,\text{enc} = 16 \cdot 128 \cdot (2 \cdot 2 + 3.5 \cdot 1) + 2 \cdot 384\,\text{op} \approx 2^{14}\,\text{op}$ (assuming 128-bit round constants).

For our attack, we use dimension $d + 1 = 55 + 1$, $n = 31444 = 2^{14.94}$ unknowns, $\bar{n} = 60320 = 2^{15.88}$ monomials. Then we select t such that $\binom{d+1+t}{d+1} \geq n$, i.e., $t = 3$: $\binom{59}{56} \approx 2^{14.99} > 2^{14.94}$. The attack recovers 337 key bits, so a single repetition is sufficient and the remaining key bits can be recovered by brute force using about $2^{384-337} = 2^{47}$ trial encryptions (or the attack can be repeated once with the same data for a different bit position of b^*). The attack procedure and complexity is then as follows:

1. **Query** the decryption oracle with a set $\mathcal{C}$ of 2^{d+1+t} chosen ciphertexts: $\mathcal{C}$ is defined by the initial structure in Figure 4a, i.e., $d+1+t$ bits of limb c enumerate all possible values, the rest is set to an arbitrary constant. Denote by $\mathcal{C}_i \leq \mathcal{C}$, $0 \leq i < 2^{d+1+t}$ the subspace of dimension $\text{wt}(i)$, where the 1-bits of i select the basis vectors of $\mathcal{C}$. Fix a selection $\{\mathcal{C}_i\}$ of n subspaces with $\text{wt}(i) = d + 1$. $(D = 2^{d+1+t})$
2. **Set up equation system**: Initialize $n \times n$ array; For each of the $\bar{n}$ monomials:
   (a) **Evaluate the monomial** for each ciphertext. $(M = D\,\text{bit}, T = \bar{n} \cdot D\,\text{op})$
   (b) **Apply the Moebius transform** to this bit vector. Extract the $n \leq \binom{d+1+t}{d}$ bits that correspond to the n subspaces $\{\mathcal{C}_i\}$ of dimension $\text{wt}(i) = d$. $(M = n\,\text{bit}, T = \bar{n} \cdot D \log_2 D\,\text{op})$
   (c) **Update equation system**: For the first n_p monomials, copy n-bit vector to an array column. For the others, update up to n_k columns. $(M = n^2\,\text{bit}, T = n_p \cdot n + (\bar{n} - n_p) \cdot n_k \cdot n\,\text{op})$
3. **Solve** system to recover the unknowns and **derive key bits**. $(T = n^3/\log n$ [Bar07]$)$

The total complexity is $D = 2^{d+1+t}$, and T may be dominated by step 1, 2b, 2c, or 3. For our parameters, this is $D = 2^{d+1+t} = 2^{59}$, $M = 2^{d+1+t} = 2^{59}\,\text{bit}$, $T \approx \bar{n} \cdot D \log_2 D = 2^{80.76}\,\text{op} \approx 2^{66.76}\,\text{enc}$.

5 Conclusion

Our analysis of Frit shows that the inverse permutation Frit^{-1} has less efficient diffusion between its 128-bit limbs than Frit, although the diffusion within each limb is much stronger. This leads to several properties that we can exploit in attacks on Frit^{-1} and, to a much lesser extent, also on Frit: First, the algebraic degree grows much more slowly over multiple rounds, with an effective degree of only $\alpha_0 \approx 1.325$. Second, by carefully selecting the variables, we can linearize up to 4 rounds of Frit^{-1} and thus obtain efficient initial structures for an integral attack. Third, we can express intermediate state bits as a polynomial in the output bits (and key bits) with a relatively limited monomial count. As a consequence, we can provide efficient attacks on the full 16-round Frit permutation if it is used as a block cipher, e.g., in an Even-Mansour mode with complexity 2^{67} when targeting the encryption, or 2^{18} when targeting the decryption. Furthermore, we provide very low-dimensional start-in-the-middle zero-sum partitioning distinguishers for the permutation.

If we consider the use of round-reduced Frit in a sponge or duplex mode of operation, the provided observations provide a good starting point for cube-like or conditional cube attacks. However, so far, we cannot exploit our observations if the full Frit permutation is used in a sponge or duplex mode of operation. Hence, we consider the analysis of these use-cases as an interesting future research topic.

Acknowledgements

We thank the Frit team for their comments on preliminary versions of the attack.

References

[ARS+15] Martin R. Albrecht, Christian Rechberger, Thomas Schneider, Tyge Tiessen, and Michael Zohner. Ciphers for MPC and FHE. In Elisabeth Oswald and Marc Fischlin, editors, *Advances in Cryptology – EUROCRYPT 2015*, volume 9056 of *LNCS*, pages 430–454. Springer, 2015.

[Bar07] Gregory V. Bard. *Algorithms for Solving Linear and Polynomial Systems of Equations over Finite Fields with Applications to Cryptanalysis.* PhD thesis, University of Maryland, College Park, MD, USA, 2007.

[BDL97] Dan Boneh, Richard A. DeMillo, and Richard J. Lipton. On the importance of checking cryptographic protocols for faults (extended abstract). In Walter Fumy, editor, *Advances in Cryptology – EUROCRYPT '97*, volume 1233 of *LNCS*, pages 37–51. Springer, 1997.

[BDP+14] Guido Bertoni, Joan Daemen, Michaël Peeters, Gilles Van Assche, and Ronny Van Keer. Keyak. Submission to the CAESAR competition: http://competitions.cr.yp.to, 2014.

[BDP+16] Guido Bertoni, Joan Daemen, Michaël Peeters, Gilles Van Assche, and Ronny Van Keer. Ketje v2. Submission to the CAESAR competition: http://competitions.cr.yp.to, 2016.

[BDPV07] Guido Bertoni, Joan Daemen, Michaël Peeters, and Gilles Van Assche. Sponge functions. Ecrypt Hash Workshop 2007, May 2007.

[BDPV08] Guido Bertoni, Joan Daemen, Michaël Peeters, and Gilles Van Assche. On the indifferentiability of the sponge construction. In Nigel P. Smart, editor, *Advances in Cryptology – EUROCRYPT 2008*, volume 4965 of *LNCS*, pages 181–197. Springer, 2008.

[BDPV11] Guido Bertoni, Joan Daemen, Michael Peeters, and Gilles Van Assche. The Keccak SHA-3 submission (Version 3.0). http://keccak.noekeon.org/Keccak-submission-3.pdf, 2011.

[BDPV12] Guido Bertoni, Joan Daemen, Michaël Peeters, and Gilles Van Assche. Duplexing the sponge: Single-pass authenticated encryption and other applications. In Ali Miri and Serge Vaudenay, editors, *Selected Areas in Cryptography – SAC 2011*, volume 7118 of *LNCS*, pages 320–337. Springer, 2012.

[BKL+17] Daniel J. Bernstein, Stefan Kölbl, Stefan Lucks, Pedro Maat Costa Massolino, Florian Mendel, Kashif Nawaz, Tobias Schneider, Peter Schwabe, François-Xavier Standaert, Yosuke Todo, and Benoît Viguier. Gimli: A cross-platform permutation. In Wieland Fischer and Naofumi Homma, editors, *Cryptographic*

Hardware and Embedded Systems – CHES 2017, volume 10529 of *LNCS*, pages 299–320. Springer, 2017.

[BS97] Eli Biham and Adi Shamir. Differential fault analysis of secret key cryptosystems. In Burton S. Kaliski Jr., editor, *Advances in Cryptology – CRYPTO '97*, volume 1294 of *LNCS*, pages 513–525. Springer, 1997.

[CCF+16] Anne Canteaut, Sergiu Carpov, Caroline Fontaine, Tancrède Lepoint, María Naya-Plasencia, Pascal Paillier, and Renaud Sirdey. Stream ciphers: A practical solution for efficient homomorphic-ciphertext compression. In Thomas Peyrin, editor, *Fast Software Encryption – FSE 2016*, volume 9783 of *LNCS*, pages 313–333. Springer, 2016.

[CCF+18] Anne Canteaut, Sergiu Carpov, Caroline Fontaine, Tancrède Lepoint, María Naya-Plasencia, Pascal Paillier, and Renaud Sirdey. Stream ciphers: A practical solution for efficient homomorphic-ciphertext compression. *Journal of Cryptology*, 31(3):885–916, 2018.

[CJRR99] Suresh Chari, Charanjit S. Jutla, Josyula R. Rao, and Pankaj Rohatgi. Towards sound approaches to counteract power-analysis attacks. In Michael J. Wiener, editor, *Advances in Cryptology – CRYPTO '99*, volume 1666 of *LNCS*, pages 398–412. Springer, 1999.

[DEG+18] Christoph Dobraunig, Maria Eichlseder, Lorenzo Grassi, Virginie Lallemand, Gregor Leander, Eik List, Florian Mendel, and Christian Rechberger. Rasta: A cipher with low anddepth and few ands per bit. In Hovav Shacham and Alexandra Boldyreva, editors, *Advances in Cryptology – CRYPTO 2018*, volume 10991 of *LNCS*, pages 662–692. Springer, 2018.

[DEM15] Christoph Dobraunig, Maria Eichlseder, and Florian Mendel. Higher-order cryptanalysis of LowMC. In Soonhak Kwon and Aaram Yun, editors, *Information Security and Cryptology – ICISC 2015*, volume 9558 of *LNCS*, pages 87–101. Springer, 2015.

[DEMS14] Christoph Dobraunig, Maria Eichlseder, Florian Mendel, and Martin Schläffer. Ascon. Submission to the CAESAR competition: http://competitions.cr.yp.to, 2014.

[DHVV18] Joan Daemen, Seth Hoffert, Gilles Van Assche, and Ronny Van Keer. Xoodoo cookbook. IACR Cryptology ePrint Archive, Report 2018/767, 2018. https://eprint.iacr.org/2018/767.

[DLMW15] Itai Dinur, Yunwen Liu, Willi Meier, and Qingju Wang. Optimized interpolation attacks on LowMC. In Tetsu Iwata and Jung Hee Cheon, editors, *Advances in Cryptology – ASIACRYPT 2015*, volume 9453 of *LNCS*, pages 535–560. Springer, 2015.

[DLWQ17] Xiaoyang Dong, Zheng Li, Xiaoyun Wang, and Ling Qin. Cube-like attack on round-reduced initialization of Ketje Sr. *IACR Transactions on Symmetric Cryptology*, 2017(1):259–280, 2017.

[DMP+15] Itai Dinur, Pawel Morawiecki, Josef Pieprzyk, Marian Srebrny, and Michal Straus. Cube attacks and cube-attack-like cryptanalysis on the round-reduced Keccak sponge function. In Elisabeth Oswald and Marc Fischlin, editors, *Advances in Cryptology – EUROCRYPT 2015*, volume 9056 of *LNCS*, pages 733–761. Springer, 2015.

[DPVR00] Joan Daemen, Michaël Peeters, Gilles Van Assche, and Vincent Rijmen. Nessie proposal: Noekeon. First Open Nessie Workshop, 2000. http://gro.noekeon.org/Noekeon-spec.pdf.

[DS09] Itai Dinur and Adi Shamir. Cube attacks on tweakable black box polynomials. In Antoine Joux, editor, *Advances in Cryptology – EUROCRYPT 2009*, volume 5479 of *LNCS*, pages 278–299. Springer, 2009.

[EM91] Shimon Even and Yishay Mansour. A construction of a cipher from a single pseudorandom permutation. In Hideki Imai, Ronald L. Rivest, and Tsutomu Matsumoto, editors, *Advances in Cryptology – ASIACRYPT '91*, volume 739 of *LNCS*, pages 210–224. Springer, 1991.

[GGNPS13] Benoît Gérard, Vincent Grosso, María Naya-Plasencia, and François-Xavier Standaert. Block ciphers that are easier to mask: How far can we go? In Guido Bertoni and Jean-Sébastien Coron, editors, *Cryptographic Hardware and Embedded Systems – CHES 2013*, volume 8086 of *LNCS*, pages 383–399. Springer, 2013.

[GLS16] Jian Guo, Meicheng Liu, and Ling Song. Linear structures: Applications to cryptanalysis of round-reduced Keccak. In Jung Hee Cheon and Tsuyoshi Takagi, editors, *Advances in Cryptology – ASIACRYPT 2016*, volume 10031 of *LNCS*, pages 249–274, 2016.

[GP99] Louis Goubin and Jacques Patarin. DES and differential power analysis (the "duplication" method). In Çetin Kaya Koç and Christof Paar, editors, *Cryptographic Hardware and Embedded Systems – CHES'99*, volume 1717 of *LNCS*, pages 158–172. Springer, 1999.

[HWX$^+$17] Senyang Huang, Xiaoyun Wang, Guangwu Xu, Meiqin Wang, and Jingyuan Zhao. Conditional cube attack on reduced-round Keccak sponge function. In Jean-Sébastien Coron and Jesper Buus Nielsen, editors, *Advances in Cryptology – EUROCRYPT 2017*, volume 10211 of *LNCS*, pages 259–288, 2017.

[JK97] Thomas Jakobsen and Lars R. Knudsen. The interpolation attack on block ciphers. In Eli Biham, editor, *Fast Software Encryption – FSE 1997*, volume 1267 of *LNCS*, pages 28–40. Springer, 1997.

[KJJ99] Paul C. Kocher, Joshua Jaffe, and Benjamin Jun. Differential power analysis. In Michael J. Wiener, editor, *Advances in Cryptology – CRYPTO '99*, volume 1666 of *LNCS*, pages 388–397. Springer, 1999.

[Koc96] Paul C. Kocher. Timing attacks on implementations of Diffie-Hellman, RSA, DSS, and other systems. In Neal Koblitz, editor, *Advances in Cryptology – CRYPTO '96*, volume 1109 of *LNCS*, pages 104–113. Springer, 1996.

[KW02] Lars R. Knudsen and David A. Wagner. Integral cryptanalysis. In Joan Daemen and Vincent Rijmen, editors, *Fast Software Encryption – FSE 2002*, volume 2365 of *LNCS*, pages 112–127. Springer, 2002.

[Lai94] Xuejia Lai. Higher order derivatives and differential cryptanalysis. In Richard E. Blahut, Daniel J. Costello Jr., Ueli Maurer, and Thomas Mittelholzer, editors, *Communications and Cryptography: Two Sides of One Tapestry*, volume 276 of *International Series in Engineering and Computer Science*, pages 227–233. Kluwer Academic Publishers, 1994.

[LBDW17] Zheng Li, Wenquan Bi, Xiaoyang Dong, and Xiaoyun Wang. Improved conditional cube attacks on Keccak keyed modes with MILP method. In Tsuyoshi Takagi and Thomas Peyrin, editors, *Advances in Cryptology – ASIACRYPT 2017*, volume 10624 of *LNCS*, pages 99–127. Springer, 2017.

[LDW17] Zheng Li, Xiaoyang Dong, and Xiaoyun Wang. Conditional cube attack on round-reduced ASCON. *IACR Transactions on Symmetric Cryptology*, 2017(1):175–202, 2017.

[LMR15] Gregor Leander, Brice Minaud, and Sondre Rønjom. A generic approach to invariant subspace attacks: Cryptanalysis of Robin, iSCREAM and Zorro. In Elisabeth Oswald and Marc Fischlin, editors, *Advances in Cryptology – EUROCRYPT 2015*, volume 9056 of *LNCS*, pages 254–283. Springer, 2015.

[MJSC16] Pierrick Méaux, Anthony Journault, François-Xavier Standaert, and Claude Carlet. Towards stream ciphers for efficient FHE with low-noise ciphertexts. In Marc Fischlin and Jean-Sébastien Coron, editors, *Advances in Cryptology – EUROCRYPT 2016*, volume 9665 of *LNCS*, pages 311–343. Springer, 2016.

[NRR06] Svetla Nikova, Christian Rechberger, and Vincent Rijmen. Threshold implementations against side-channel attacks and glitches. In Peng Ning, Sihan Qing, and Ninghui Li, editors, *Information and Communications Security – ICICS 2006*, volume 4307 of *LNCS*, pages 529–545. Springer, 2006.

[NRS08] Svetla Nikova, Vincent Rijmen, and Martin Schläffer. Secure hardware implementation of non-linear functions in the presence of glitches. In Pil Joong Lee and Jung Hee Cheon, editors, *Information Security and Cryptology – ICISC 2008*, volume 5461 of *LNCS*, pages 218–234. Springer, 2008.

[NRS11] Svetla Nikova, Vincent Rijmen, and Martin Schläffer. Secure hardware implementation of nonlinear functions in the presence of glitches. *Journal of Cryptology*, 24(2):292–321, 2011.

[RASA14] Shahram Rasoolzadeh, Zahra Ahmadian, Mahmoud Salmasizadeh, and Mohammad Reza Aref. Total break of Zorro using linear and differential attacks. IACR Cryptology ePrint Archive, Report 2014/220, 2014.

[SBD$^+$18] Thierry Simon, Lejla Batina, Joan Daemen, Vincent Grosso, Pedro Maat Costa Massolino, Kostas Papagiannopoulos, Francesco Regazzoni, and Niels Samwel. Towards lightweight cryptographic primitives with built-in fault-detection. IACR Cryptology ePrint Archive, Report 2018/729, 2018.

[SLG17] Ling Song, Guohong Liao, and Jian Guo. Non-full sbox linearization: Applications to collision attacks on round-reduced Keccak. In Jonathan Katz and Hovav Shacham, editors, *Advances in Cryptology – CRYPTO 2017*, volume 10402 of *LNCS*, pages 428–451. Springer, 2017.

[SMG16] Tobias Schneider, Amir Moradi, and Tim Güneysu. ParTI – towards combined hardware countermeasures against side-channel and fault-injection attacks. In Matthew Robshaw and Jonathan Katz, editors, *Advances in Cryptology – CRYPTO 2016*, volume 9815 of *LNCS*, pages 302–332. Springer, 2016.

[Tod15a] Yosuke Todo. Integral cryptanalysis on full MISTY1. In Rosario Gennaro and Matthew Robshaw, editors, *Advances in Cryptology – CRYPTO 2015*, volume 9215 of *LNCS*, pages 413–432. Springer, 2015.

[Tod15b] Yosuke Todo. Structural evaluation by generalized integral property. In Elisabeth Oswald and Marc Fischlin, editors, *Advances in Cryptology – EUROCRYPT 2015*, volume 9056 of *LNCS*, pages 287–314. Springer, 2015.

[WWGY14] Yanfeng Wang, Wenling Wu, Zhiyuan Guo, and Xiaoli Yu. Differential cryptanalysis and linear distinguisher of full-round Zorro. In Ioana Boureanu, Philippe Owesarski, and Serge Vaudenay, editors, *Applied Cryptography and Network Security – ACNS 2014*, volume 8479 of *LNCS*, pages 308–323. Springer, 2014.

A Symbolic Evaluation with Sage

Listing 1: Sage code to count the monomials in limb b after $1, \ldots, 5$ rounds of $\textsc{Frit}^{-1}$.

```
varnames = sum([[v+str(i) for i in range(384)] for v in 'CK'], [])
R = BooleanPolynomialRing(len(varnames), varnames, order='deglex')
Cvar, Kvar = list(R.gens())[:384], list(R.gens())[384:]

AND  = lambda x, y: [xi*yi for xi, yi in zip(x, y)]
XOR  = lambda x, y: [xi+yi for xi, yi in zip(x, y)]
XOR3 = lambda x, y, z: [xi+yi+zi for xi, yi, zi in zip(x, y, z)]
ROTL = lambda x, r: x[-r:] + x[:-r]
invlist = lambda l: [r for r, xr in enumerate(matrix.circulant(
                      [GF(2)(1) if i in l else GF(2)(0) for i in range(128)]
                    ).inverse()[0]) if xr]
SIGMA = lambda x, l: reduce(lambda x, y: XOR(x, y), [ROTL(x, li) for li in l])
SIGMA_a = lambda x: XOR3(x, ROTL(x, 110), ROTL(x, 87))
SIGMA_c = lambda x: XOR3(x, ROTL(x, 118), ROTL(x, 88))
SIGMA_a_inv = lambda x: SIGMA(x, invlist([0,87,110]))
SIGMA_c_inv = lambda x: SIGMA(x, invlist([0,88,118]))
RC = [GF(2)(rc) for rc in reversed(0xF9A42BB1.binary().zfill(128))]
RCs = [i*[GF(2)(0)] + RC[i:] for i in range(16)]

texnum = lambda x: str(x) + "␣=␣2^{" + str(log(x,2).n(digits=5)) + "}"
def STATS(rnd, x):
  print "ROUND", rnd
  print "F[K,C]:",[len(x.graded_part(d)) for d in [0..8]],"=",texnum(len(x))
  subsdict = {ki:R.one() for ki in Kvar}
  x = sum(set([mon.subs(subsdict) for mon in x.monomials()]))
  print "F_K[C]:",[len(x.graded_part(d)) for d in [0..8]],"=",texnum(len(x))

nrounds = 5
a = XOR(Cvar[0:128],   Kvar[0:128])
b = XOR(Cvar[128:256], Kvar[128:256])
c = XOR(Cvar[256:384], Kvar[256:384])

c = XOR3(c, AND(a, b), RCs[0])
a = SIGMA_a_inv(a)
STATS("1b", b[0])

for r in [2..(nrounds-2)]:
  c, a, b = a, b, c
  b = XOR3(b, a, c)
  c = XOR3(SIGMA_c_inv(c), AND(a, b), RCs[r-1])
  a = SIGMA_a_inv(a)
  STATS(str(r)+"b", b[0])

c, a, b = a, b, c
b = XOR3(b[:1], a[:1], c[:1])
STATS(str(nrounds-1)+"b", b[0])
b0 = (SIGMA_a_inv(a)[0] + b[0] + SIGMA_c_inv(c)[0]) + b[0] * a[0]
STATS(str(nrounds)+"b", b0)
```