The QARMA Block Cipher Family

Almost MDS Matrices Over Rings With Zero Divisors, Nearly Symmetric Even-Mansour Constructions With Non-Involutory Central Rounds, and Search Heuristics for Low-Latency S-Boxes

Roberto Avanzi

Qualcomm Product Security, Munich, Germany
ravanzi@qti.qualcomm.com, roberto.avanzi@gmail.com

Abstract. This paper introduces QARMA, a new family of lightweight tweakable block ciphers targeted at applications such as memory encryption, the generation of very short tags for hardware-assisted prevention of software exploitation, and the construction of keyed hash functions.

QARMA is inspired by reflection ciphers such as PRINCE, to which it adds a tweaking input, and MANTIS. However, QARMA differs from previous reflector constructions in that it is a *three-round Even-Mansour scheme* instead of a FX-construction, and its middle permutation is *non-involutory and keyed*. We introduce and analyse a family of Almost MDS matrices defined over a ring with zero divisors that allows us to encode rotations in its operation while maintaining the minimal latency associated to $\{0, 1\}$-matrices. The purpose of all these design choices is to harden the cipher against various classes of attacks.

We also describe new S-Box search heuristics aimed at minimising the critical path.

QARMA exists in 64- and 128-bit block sizes, where block and tweak size are equal, and keys are twice as long as the blocks.

We argue that QARMA provides sufficient security margins within the constraints determined by the mentioned applications, while still achieving best-in-class latency.

Implementation results on a state-of-the art manufacturing process are reported.

Finally, we propose a technique to extend the length of the tweak by using, for instance, a universal hash function, which can also be used to strengthen the security of QARMA.

Keywords: Tweakable Block Ciphers · Reflection Ciphers · Even-Mansour Schemes · Almost MDS Matrices · S-Box Search Heuristics · Memory Encryption · Pointer Authentication · Short Hashes · Tweak Masking · Tweak Extension

IACR Transactions on Symmetric Cryptology Vol. 0, No.0, pp.1—41, DOI:XXXXXXXX

Contents

1 Introduction **3**

2 Specification of QARMA **5**
2.1 General Definitions and Notation 5
2.2 Key Specialisation 6
2.3 The Forward Round Function 6
2.4 The Tweak Update Function 6
2.5 The Backward Round Function 6
2.6 The Central Construction and the Pseudo-Reflector 6
2.7 The Encryption and Decryption Algorithm 7
2.8 Parameters 7
2.9 Usage, Modes of Operation, Tweak Extension 8

3 Design Decisions **8**
3.1 The Diffusion Matrices 8
3.2 The Central Construction 15
3.3 Selection of the 4-Bit S-Boxes 15
3.4 The 8-bit S-Boxes 19
3.5 The ω Function 20
3.6 The Round Constants 21

4 Security Analysis **21**
4.1 Common Attacks on Block Ciphers 21
Linear and Differential Cryptanalysis 21
Slide Attacks 21
Impossible Differential and Zero-Correlation Linear Cryptanalysis 22
Boomerang, Integral and Meet-in-the-Middle Attacks 22
4.2 Algebraic Attacks 22
4.3 Invariant Subspace Cryptanalysis 23
4.4 Security Implications of the Central Construction 25
4.5 Attacks on Even-Mansour Schemes 26
4.6 Security Claims 27
4.7 Security for Specific Applications 28

5 Hardware Implementation **28**

6 Conclusions and Open Questions **31**

Acknowledgements **31**

References **32**

A Test Vectors **38**
A.1 QARMA-64 with $M = Q = M_{4,2}$ 38
A.2 QARMA-128 with $M = Q = M_{8,2}$ 38
A.3 QARMA-64 with $M = Q = M_{4,1}$ (legacy) 39
A.4 QARMA-128 with $M = M_{8,4}$ and $Q = Q_{8,4}$ (legacy) 39

B Alternative Tweak Masking Function **40**

C Proof that $M_{8,2}$ has 2^{72} Fixed Points on the State **41**

1 Introduction

Research on lightweight block ciphers is motivated by the need to provide an acceptable security level for *specific* applications at much lower area, latency, or power consumption levels than, say, the `AES` [DR02]. For lightweight ciphers, all components must be tightly optimised, often leading to original solutions, as embodied by designs like `CLEFIA` [SSA+07], `KATAN` [CDK09], `KLEIN` [GNL11], `LED` [GPPR12], `PRESENT` [BKL+07], `PRINCE` [BCG+12], `SIMON` and `SPECK` [BSS+13], and `MIDORI` [BBI+15], to name just a few.

At the same time, research on *tweakable* block ciphers [LRW02] (TBC) is motivated by the design of encryption modes of operation and hash functions for specific applications, such as disk [Mar10] and memory encryption [HT13]. In addition to the secret key and a plaintext or ciphertext, TBCs accept a *third* input, the tweak, which is public. The tweak and key together select the permutation computed by the cipher.

To understand the rationale behind the research into tweakable encryption primitives, let us consider the problem of memory confidentiality. The simplest encryption mode is the Electronic Code-Book mode (ECB), whereby each block is simply encrypted with the same key. However, ECB preserves a lot of the structure of the plaintext in the ciphertext, making it unsuitable for applications that require a high level of confidentiality. Among alternatives that hide the structure of the plaintext, for performance reasons a parallelisable mode is preferred. A common choice is represented by counter based modes, which turn the block cipher into, essentially, a stream cipher, as used in AEGIS [SCG+03], in [YEP+06], and in Intel's SGX [Gue16]. However, these modes require extra memory for the nonces and are difficult to properly implement in several contexts since they need a good source of entropy. If nonces are reused, these modes break completely.

In order to be able to target constrained devices, we see then that it is desirable to have memory encryption primitives that provide some level of nonce misuse resistance, or can even work without any memory expansion. We consider modes that employ direct encryption, where the permutation depends on a tweak that can be the physical address of the block being encrypted, and/or an additional nonce or counter if memory expansion is allowed. The dependency can be realised in the mode of operation (around a non-tweakable block cipher) or in the underlying cipher. In the latter case we have a TBC.

The first TBC, the `AES` submission *Hasty Pudding Cipher* [Sch99], was quickly followed by schemes to create TBCs from ordinary primitives used as black boxes [Rog04, LST12, ST13], the cipher `MERCY` [Cro00], the cipher `THREEFISH` at the core of the `SKEIN` hash function [FLS+10], and the ciphers `Deoxys-BC`, `Joltik-BC`, and `Kiasu-BC` based on the `TWEAKEY` framework [JNP14]. More recently, we have `SKINNY` and `MANTIS` [BJK+16].

Unfortunately, most generic constructions to add a tweak to a block cipher or to a mode of encryption suffer from vastly increased latency. For instance, the XEX [Rog04] mode of encryption requires an additional encryption operation before the plaintext is actually encrypted. The deployment of a TBC with *latency comparable to a usual block cipher* would therefore improve performance, as long as *changing the tweak is inexpensive*.

A further application of TBCs is to software security, to enforce code flow integrity (CFI) by inserting short tags into unused bits of pointers [ARM16, QPS17]. These tags depend on a secret key and on a public value associated to the pointer's context. Since the latter may change often, a TBC is an obvious candidate to implement a CPU instruction to compute the tags. Since instructions with higher latency inject a longer "bubble" into the processing pipeline in critical places of the code such as function prologue and epilogue, reduction of the total latency is critical for this use case as well.

For all mentioned applications area and energy consumption are secondary parameters with respect to reduction of the total latency: For memory and disk encryption the energy consumption is dominated by that of the memory or mass storage controller and related hardware; in a modern CPU a few thousand gates are often nearly negligible.

A tweakable cipher is secure if it cannot be broken even assuming the adversary has

full tweak control. Hence, designing a TBC is a difficult task as care must be taken in how the user-controlled tweak is included in the design.

Contents of the Paper. In this work, we propose QARMA (pronounced like the sanskrit word *karma*), a family of hardware-oriented lightweight TBCs. QARMA is chiefly meant to be deployed in fully unrolled or pipelined hardware implementations. In particular, its design is not intended to facilitate round-based implementations. QARMA targets use cases such as memory encryption, and short tags for software security.

QARMA is a bricklayer SPN. It reuses some concepts from PRINCE, MIDORI and MANTIS, but there are important differences both in structure and in choice of components. The common aspects allow us to focus our analysis on the new ones. The main differences are:

1. QARMA's structure is an Even-Mansour scheme [EM91, EM97] with a keyed *pseudo-reflector* to avoid certain types of cryptanalysis, a departure from the more common FX-construction [Rog96] used in other reflector designs.

2. To define QARMA's diffusion layer, we introduce and analyse a family of Almost MDS matrices defined over the ring with zero divisors $R_m = \mathbb{F}_2[\rho] = \mathbb{F}_2[X]/(X^m + 1)$. This allows us to encode circular rotations in its operation while maintaining the minimal latency associated to $\{0, 1\}$-matrices. To the best of our knowledge this is the first work addressing the characterization of Almost MDS circulant matrices over this ring. Such matrices help hedge against some types of iterative characteristics, invariant subspace attacks, and can be used to construct better central rounds for reflection ciphers. We also characterise those matrices with equally efficient inverses.

3. We introduce new heuristics to efficiently find S-Boxes with short critical path.

4. The TWEAKEY framework is taken as an inspiration: The bits of key and tweak are not permuted synchronously, but instead only those of the tweak are shuffled between rounds; Additionally, a LFSR is used to update the tweak.

QARMA's latency is sufficiently small to allow usage in a tweaked ECB mode that eschews the expensive masking value derivation typical of XEX-like constructions. Absolute minimisation of area is not our primary design goal, so we allow some more expensive choices to build better security margins.

There are two variants of QARMA that support blocks sizes of $n = 64$ and $n = 128$ bits, denoted by QARMA-64 and QARMA-128, respectively. The tweak is also n bits long and the key is always $2\,n$ bits long. QARMA-128 is suitable for higher security memory or storage encryption. We do not consider the fact that QARMA-128 only supports 256-bit keys a limitation; in fact the larger key still gives 128 bits of security under Grover's algorithm [Gro96], making it preferable over QARMA-64 if post-quantum resistance is desired. Also note that for storage or memory encryption the use of larger keys is common: for instance, if AES-128 in XTS-mode is used, 256 bits of key material are *already* required to meet NIST FIPS compliance requirements [NIS16, Appendix A.9]. Hence, the same key storage structures can be reused if QARMA-128 replaces or supplements the AES.

The security of QARMA is thoroughly investigated. We also report on implementation results with a state-of-the-art FinFet 7 nm manufacturing process.

A further contribution of this paper is the proposal of a simple technique to extend the length of the tweak of a block cipher that has only a fixed length tweak. The idea consists in using a universal hash function (such as a multi-linear UHF) to compress a longer tweak onto the available size. This idea may seem counterintuitive, as it introduces an additional computational cost to change the tweak. However, in practice the resulting construction may be more efficient than increasing the number of rounds of the cipher to include a longer tweak while maintaining the same security level. While this line of research seems promising, it needs more analysis.

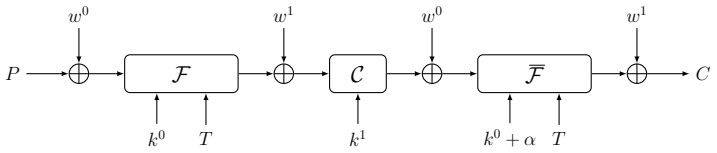

Figure 1: The Overall Scheme

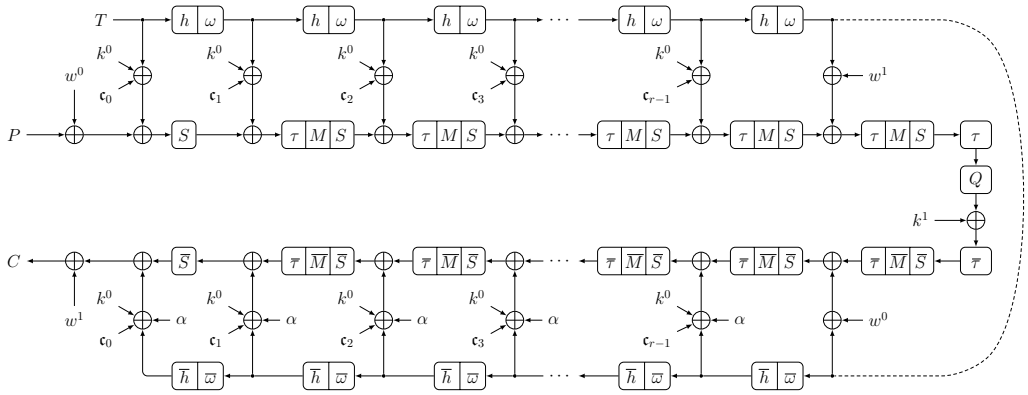

Figure 2: The Structure of $\texttt{QARMA}_r$

2 Specification of QARMA

2.1 General Definitions and Notation

The overall scheme of the TBC QARMA is depicted in Figure 1. There, and throughout the paper, a bar over a function – e.g. $\overline{F}$ – denotes its inverse. QARMA is a three-round Even-Mansour construction where the permutations are parameterized by a core key, and the key mixings between rounds are derived from a whitening key. The first and third permutations are functionally the inverse of each other and are further parameterized by a tweak. The central permutation is designed to be easily inverted by means of a simple transformation of the key.

The cipher is depicted in more detail in Figure 2. Both similarities and differences with respect to previous designs can be clearly seen from the picture.

The keys k^0, k^1, w^0, and w^1 are derived from a master key K via a simple *key specialisation*. The letters P, C and T denote the plaintext, the ciphertext and the tweak; S represents a layer of sixteen m-bit S-Boxes, h and τ are permutations, M and Q are MixColumns-like operations, with Q involutory, and ω is a LFSR.

Write $n = 16\,m$ with $m = 4$ or 8. All n-bit values are represented as arrays of sixteen m-bit *cells*. Cells are indexed in big endian order (hence, for QARMA-64, bits 63..60 are contained in the zeroth cell, and bits 3..0 in in the fifteenth cell) while the bits inside a cell are ordered in little endian order. Any array of sixteen cells is also viewed as a 4×4 matrix, for instance, the internal state admits representations

$$\mathrm{IS} = s_0 \| s_1 \| \cdots \| s_{14} \| s_{15} = \begin{pmatrix} s_0 & s_1 & s_2 & s_3 \\ s_4 & s_5 & s_6 & s_7 \\ s_8 & s_9 & s_{10} & s_{11} \\ s_{12} & s_{13} & s_{14} & s_{15} \end{pmatrix} , \tag{1}$$

so that 4×4 matrices operate column-wise on these values by left multiplication. The plaintext is given as $P = p_0 \| p_1 \| \cdots \| p_{14} \| p_{15}$, the tweak as $T = t_0 \| t_1 \| \cdots \| t_{14} \| t_{15}$.

Throughout the paper, we use the symbol "+" to denote addition in all algebraic structures. In particular it denotes the exclusive or in the QARMA ciphers, which do not use modular addition. The symbol tk denotes a (round) *tweakey*, i.e. a value derived only from the key, the tweak, and the round constants.

2.2 Key Specialisation

The $2n = 32m$-bit key K is first partitioned as $w^0 \| k^0$ where w^0 and k^0, the *whitening* and *core* keys, are $16m$ bits each.

For encryption, put $w^1 = o(w^0) := (w^0 \ggg 1) + (w^0 \gg (16m - 1))$ and $k^1 = k^0$.

Since the first r rounds of the cipher (ignoring initial whitening) differ from last r rounds solely by the addition of a non-zero constant α, QARMA possesses a property very similar to PRINCE's α-reflection: The encryption circuit can be used for decryption when $k^0 + \alpha$ is used as the core key, the whitening keys w^0 with w^1 are swapped, and $k^1 = Q \cdot k^0$.

2.3 The Forward Round Function

The *Forward Round Function* $\mathcal{R}(\mathrm{IS}; \mathsf{tk})$ is composed by four operations, performed in the following order:

1. AddRoundTweakey. The round tweakey tk defined in § 2.7 is XORed to IS.
2. ShuffleCells. $(\tau(\mathrm{IS}))_i = s_{\tau(i)}$ for $0 \leq i \leq 15$, where τ is the MIDORI cell permutation, i.e. $\tau = [\,0, 11, 6, 13, 10, 1, 12, 7, 5, 14, 3, 8, 15, 4, 9, 2\,]$.
3. MixColumns. Each column of the cipher internal state array is multiplied by the matrix M defined in § 3.1, i.e. $\mathrm{IS} = M \cdot \mathrm{IS}$.
4. SubCells. For the chosen S-Box σ, the S layer acts on the state as follows: $s_i \leftarrow\!\shortmid\, \sigma(s_i)$ for $0 \leq i \leq 15$. The S-Boxes are defined in §§ 3.3 and 3.4.

A *short* version of the forward round function exists which omits the ShuffleCells and MixColumns operations, similarly to the AES final round.

After AddRoundTweakey the tweak T is updated by the function described next.

2.4 The Tweak Update Function

First, the cells of the tweak are permuted as $h(T) = t_{h(0)} \| \cdots \| t_{h(15)}$, where h is the same permutation $h = [\,6, 5, 14, 15, 0, 1, 2, 3, 7, 12, 13, 4, 8, 9, 10, 11\,]$ used in MANTIS.

Then, a LFSR ω updates the tweak cells with indexes 0, 1, 3, 4, 8, 11, and 13. For $m = 4$, ω is a maximal period LFSR that maps cell (b_3, b_2, b_1, b_0) to $(b_0 + b_1, b_3, b_2, b_1)$. For $m = 8$, it maps cell $(b_7, b_6, \ldots, b_0)$ to $(b_0 + b_2, b_7, b_6, \ldots, b_1)$, and its cycles on the non-zero values have all length 15 or 30.

2.5 The Backward Round Function

The *Backward Round Function* $\overline{\mathcal{R}}(\mathrm{IS}; \mathsf{tk})$ is the inverse of the forward round function $\mathcal{R}$. Its *short* form omits ShuffleCells and MixColumns. The tweak update using the inverse LFSR $\overline{\omega}$ and the inverse permutation $\overline{h}$ must be applied before AddRoundTweakey.

2.6 The Central Construction and the Pseudo-Reflector

Two *central rounds* – a forward and a backward one – that use the whitening key instead of the core key, bracket the cipher's *Pseudo-Reflector* $\mathcal{P}(\mathrm{IS}; \mathsf{tk})$, which is essentially just a key addition and a matrix multiplication of the internal state. In more detail, this central construction is defined as follows:

Algorithm QARMA$_r$ Encryption/Decryption

1: Write $K = w^0 \| k^0$
2: If **ENC**: $w^1 \leftarrow o(w^0)$, $k^1 \leftarrow k^0$
   If **DEC**: $w^1 \leftarrow w^0$, $w^0 \leftarrow o(w^0)$, $k^1 \leftarrow Q \cdot k^0$, $k^0 \mapsto k^0 + \alpha$
3: IS $\leftarrow P + w^0$
4: for $i = 0$ to $r - 1$ do
5: IS $\leftarrow \mathcal{R}(\text{IS}, k^0 + T + \mathsf{c}_i)$ (short round for $i = 0$)
6: $T \leftarrow \omega \circ h(T)$
7: IS $\leftarrow \mathcal{R}(\text{IS}, w^1 + T)$
8: IS $\leftarrow \mathcal{P}(\text{IS}, k^1)$
9: IS $\leftarrow \mathcal{R}(\text{IS}, w^0 + T)$
10: for $i = r - 1$ down to 0 do
11: $T \leftarrow \overline{h} \circ \overline{\omega}(T)$
12: IS $\leftarrow \overline{\mathcal{R}}(\text{IS}, k^0 + T + \mathsf{c}_i + \alpha)$ (short round for $i = 0$)
13: $C \leftarrow \text{IS} + w^1$

Figure 3: The QARMA Algorithm

1. A forward round $\mathcal{R}$.
2. The pseudo-reflector $\mathcal{P}(\text{IS}; \mathsf{tk})$ i.e.
   (a) ShuffleCells.
   (b) Multiplication of the state by the involutory matrix Q defined in § 3.1.
   (c) AddRoundTweakey. The round tweakey tk is XORed to the state.
   (d) Inverse ShuffleCells.
3. A backward round $\overline{\mathcal{R}}$.

It is clear that if steps (b) and (c) were swapped, then tk would have to be replaced with $\overline{Q} \cdot \mathsf{tk} = Q \cdot \mathsf{tk}$ to obtain the same function. Because of this, if tk is the tweakey used during encryption, $Q \cdot \mathsf{tk}$ must be used instead to decrypt.

2.7 The Encryption and Decryption Algorithm

The encryption algorithm of QARMA$_r$ is given in Figure 3. QARMA$_r$ has $2\,r + 2$ rounds.

The round constants are derived from the expansion of the constant π. For the 64-bit version of QARMA we replace the first block of sixteen digits of the fractional part with zeros and select the seventh block as the α constant, as shown in Table 1 – as a *hommage* to PRINCE. For the 128-bit cipher, instead, we just take the first block of 128 bits in the fractional part of π as the α constant, set $\mathsf{c}_0 = 0$, and then each c_i is a successive block 128 bits of π, as shown in Table 2.

Note the constant α is always added to the last r backward rounds. This and the pseudo-reflector design prevent perfect symmetry in the data obfuscation path.

2.8 Parameters

For QARMA$_r$-64 we choose $r = 7$, i.e. 16 rounds, but we believe the cipher to be safe against practical attacks already for $r = 6$, i.e. 14 rounds, with some use cases even allowing for $r = 5$, i.e. 12 rounds. For QARMA$_r$-128 we choose $r = 11$, i.e. 24 rounds, and believe the cipher safe against practical attacks already for $r = 8$, i.e. 18 rounds. We shall argue in Section 4 that these parameters offer sufficient security margins.

Table 1: The Round Constants for the 64-bit Ciphers

α	=	C0AC29B7C97C50DD	$\mathfrak{c}_0$	=	0000000000000000	$\mathfrak{c}_1$	=	13198A2E03707344
$\mathfrak{c}_2$	=	A4093822299F31D0	$\mathfrak{c}_3$	=	082EFA98EC4E6C89	$\mathfrak{c}_4$	=	452821E638D01377
$\mathfrak{c}_5$	=	BE5466CF34E90C6C	$\mathfrak{c}_6$	=	3F84D5B5B5470917	$\mathfrak{c}_7$	=	9216D5D98979FB1B ...

Table 2: The Round Constants for the 128-bit Ciphers

α	=	243F6A8885A308D3 13198A2E03707344	$\mathfrak{c}_0$	=	0000000000000000 0000000000000000
$\mathfrak{c}_1$	=	A4093822299F31D0 082EFA98EC4E6C89	$\mathfrak{c}_2$	=	452821E638D01377 BE5466CF34E90C6C
$\mathfrak{c}_3$	=	C0AC29B7C97C50DD 3F84D5B5B5470917	$\mathfrak{c}_4$	=	9216D5D98979FB1B D1310BA698DFB5AC
$\mathfrak{c}_5$	=	2FFD72DBD01ADFB7 B8E1AFED6A267E96	$\mathfrak{c}_6$	=	BA7C9045F12C7F99 24A19947B3916CF7
$\mathfrak{c}_7$	=	0801F2E2858EFC16 636920D871574E69	$\mathfrak{c}_8$	=	A458FEA3F4933D7E 0D95748F728EB658
$\mathfrak{c}_9$	=	718BCD5882154AEE 7B54A41DC25A59B5	$\mathfrak{c}_{10}$	=	9C30D5392AF26013 C5D1B023286085F0
		...			

2.9 Usage, Modes of Operation, Tweak Extension

QARMA is intended to be used with a single key per security domain, application, and context/session, and with a variable tweak that is usually a public token of information.

For memory encryption methods that require dependency on both the physical address A and a nonce/counter ν, QARMA-128 offers sufficient entropy in the tweak: A and ν can usually be just concatenated and padded. This is often not possible with QARMA-64, since the tweak is only 64 bits long. In this case the tweak for QARMA-64 can be computed as $\beta \cdot A + \gamma \cdot \nu$ in some arithmetic structure – e.g. a Galois field or ring of modular integers – where β and γ are also secret values, that is, additional key material. This also leads to the observation that QARMA can also be used in a more conservative mode by multiplying the tweak by a second secret key.

This technique can be viewed as a generic method to extend the length of the tweak beyond n bits for QARMA-n, using a multi-linear universal hash function. We remark that the loss in latency due to the multiplications may be smaller than with the additional number of rounds that would be required to mix in more tweak material in a secure way. In fact, in [BJK+16] the versions of the cipher SKINNY-n with tweakeys of length $2\,n$, resp. $3\,n$ require up to 20%, 40% more rounds than the versions with n-bit tweakeys. Looking at the implementation results in Section 5, similar increases in the number of rounds would usually exceed the latency of, say, a Galois Multiplication.

Moreover, in place of Galois or modular multiplication other, lighter functions can be used that guarantee good diffusion of the bits of the tweak and of the secret value(s). Such a function does not have to be highly nonlinear. An admissible candidate is the "fresh re-keying" function described in [MSGR10]. Another one is presented in Appendix B.

3 Design Decisions

We now describe how we chose the building blocks and structure of QARMA. This section is a "flattening" of the non-linear, often iterative process that led to these choices – there are by necessity a few forward references, including to the security analysis (Section 4).

3.1 The Diffusion Matrices

3.1.1 Characterisation

QARMA's diffusion layer is composed of a cell permutation and a matrix multiplication. Its complexity mostly comes from the matrix. Roughly speaking, the usual requirements on a diffusion matrix are:

1. It should guarantee mathematically provable good diffusion.

2. It should be as lightweight as possible.

Regarding the first requirement, diffusion is usually measured by the branch number [DR02], i.e. the smallest nonzero joint number of active inputs and outputs of the matrix. For an invertible $s \times s$ matrix M, the branch number $\mathcal{B}_M$ cannot be greater than $s+1$. The MDS (Maximum-Distance Separable) matrices are those that attain this maximum: They have long been the preferred choice of block cipher designers, but they tend to lead to expensive implementations. Hence there has been a recent flurry of research on lighter diffusion layers, culminating with those of `PRINCE`, `PRIDE` [ADK+14] and `MIDORI`. In particular in `MIDORI` a $\{0,1\}$-matrix M – namely a matrix whose entries are in $\{0,1\}$ – is used that is *Almost MDS*, i.e. with $\mathcal{B}_M = s$. Almost MDS matrices have worse diffusion than MDS matrices, but they can be much lighter, so that even if more rounds are necessary to attain the same level of security than in a design with MDS matrices, the resulting cipher can still be smaller and faster.

The second requirement is often understood as *minimising the weight of the matrix*, but for fully unrolled HW implementations a better statement would be minimising the maximum of the weights of all the rows, taking into account the weights of the cells, and the underlying algebraic structure.

From this point of view, $\{0,1\}$-matrices are clearly optimal. They are never MDS, but they are Almost MDS in dimensions 2, 3, and 4 [CK08], and only in these dimensions. However, they may also contribute to the weaknesses of some ciphers: In the case of `MIDORI`, they permit the propagation of iterative characteristics that are built from the structure of the set of fixed points of the S-Box. In order to harden against attacks that exploit these properties it is desirable to have diffusion layers that do not help propagate such high likelihood characteristics. More in detail, assuming we are using a single type of S-Box, if the diffusion layer carries the sum of the outputs of two active S-Boxes unchanged into two different target cells, these cells will carry over the same characteristic twice into the next S-Box layer: the second copy of that characteristic will come for free.

The intuition is that if the S-Box outputs are subject to different linear transformations per each target cell before being added, then the two resulting output characteristics will be different, and at least one will be less likely to propagate through the next S-Box layer. Hence, we look beyond $\{0,1\}$-matrices. Usually, this means looking at matrices over binary extension fields.

A problem with matrices over binary extension fields is that any multiplication by an element different from 0 and 1 requires a modular reduction step (for a polynomial basis) or expansion step (for a normal basis) which adds latency. So the next logical step is to consider a different underlying algebraic structure, for instance a quotient ring $R_m = \mathbb{F}_2[X]/(X^m+1)$. The multiplication by the image ρ of X in the ring R_m (an element such that $\rho^m = 1$, and thus such that $\{1, \rho, \rho^2, \ldots, \rho^{m-1}\}$ is a basis for R_m as a $\mathbb{F}_2$-algebra) is just a simple circular left rotation of the bits, with only signal propagation latency. Matrices over R_m allow us to easily include rotations in the diffusion layer.

It turns out that, while it seems difficult to construct MDS matrices over generic rings, with a little effort we can find Almost MDS matrices over quotient rings $R_m = \mathbb{F}_2[X]/(X^m+1)$. Since R_m contains zero divisors (for $m \geq 2$), care shall be taken when constructing invertible matrices. We thus restrict ourselves to the following circulants:

$$M = \operatorname{circ}(0, \rho^a, \rho^b, \rho^c) = \begin{pmatrix} 0 & \rho^a & \rho^b & \rho^c \\ \rho^c & 0 & \rho^a & \rho^b \\ \rho^b & \rho^c & 0 & \rho^a \\ \rho^a & \rho^b & \rho^c & 0 \end{pmatrix} . \tag{2}$$

Theorem 1. *Let $R_m = \mathbb{F}_2[\rho]$ be the quotient ring $\mathbb{F}_2[X]/(X^m + 1)$ where ρ is the image of X in R, $m \geq 2$. The Almost MDS matrices M of the form (2) over the ring R_m are precisely the invertible ones, i.e. are those for which $\rho^{4a} + \rho^{4b} + \rho^{4c} \in R_m^*$.*

Proof. Since $\det(M) = \rho^{4a} + \rho^{4b} + \rho^{4c}$, the invertible matrices are those for which this value is invertible, i.e. in R^*. Since Almost MDS matrices are invertible by definition, we need only prove that any M of form (2) has branch number four.

Let us consider a column vector $V = (v_0, v_1, v_2, v_3)^t$. We need to verify when the sum of the weights of $V \neq 0$ and $U := M \cdot V$ is always at least four, so we need to consider the cases where V has weights one, two, and three. Since M is circulant, there are only four distinct cases up to circular permutation of the entries of V:

(i) $V = (v_0, 0, 0, 0)^t$ with $v_0 \neq 0$. Then

$$U = M \cdot V = \begin{pmatrix} 0 & \rho^a & \rho^b & \rho^c \\ \rho^c & 0 & \rho^a & \rho^b \\ \rho^b & \rho^c & 0 & \rho^a \\ \rho^a & \rho^b & \rho^c & 0 \end{pmatrix} \cdot \begin{pmatrix} v_0 \\ 0 \\ 0 \\ 0 \end{pmatrix} = \begin{pmatrix} 0 \\ v_0\,\rho^c \\ v_0\,\rho^b \\ v_0\,\rho^a \end{pmatrix}$$

and since ρ is not a zero divisor in R, the vector U has weight 3.

(ii) $V = (v_0, v_1, 0, 0)^t$ with v_0, $v_1 \neq 0$. Then

$$U = M \cdot V = \begin{pmatrix} 0 & \rho^a & \rho^b & \rho^c \\ \rho^c & 0 & \rho^a & \rho^b \\ \rho^b & \rho^c & 0 & \rho^a \\ \rho^a & \rho^b & \rho^c & 0 \end{pmatrix} \cdot \begin{pmatrix} v_0 \\ v_1 \\ 0 \\ 0 \end{pmatrix} = \begin{pmatrix} v_1\,\rho^a \\ v_0\,\rho^c \\ v_0\,\rho^b + v_1\,\rho^c \\ v_0\,\rho^a + v_1\,\rho^b \end{pmatrix}$$

and the first two entries of U are nonzero.

(iii) $V = (v_0, 0, v_2, 0)^t$ with v_0, $v_2 \neq 0$. Then

$$U = M \cdot V = \begin{pmatrix} 0 & \rho^a & \rho^b & \rho^c \\ \rho^c & 0 & \rho^a & \rho^b \\ \rho^b & \rho^c & 0 & \rho^a \\ \rho^a & \rho^b & \rho^c & 0 \end{pmatrix} \cdot \begin{pmatrix} v_0 \\ 0 \\ v_2 \\ 0 \end{pmatrix} = \begin{pmatrix} v_2\,\rho^b \\ v_0\,\rho^c + v_2\,\rho^a \\ v_0\,\rho^b \\ v_0\,\rho^a + v_2\,\rho^c \end{pmatrix}$$

and there is nothing to prove also in this case.

(iv) $V = (v_0, v_1, v_2, 0)^t$ with v_0, v_1, and $v_2 \neq 0$. Then

$$U = M \cdot V = \begin{pmatrix} 0 & \rho^a & \rho^b & \rho^c \\ \rho^c & 0 & \rho^a & \rho^b \\ \rho^b & \rho^c & 0 & \rho^a \\ \rho^a & \rho^b & \rho^c & 0 \end{pmatrix} \cdot \begin{pmatrix} v_0 \\ v_1 \\ v_2 \\ 0 \end{pmatrix} = \begin{pmatrix} v_1\,\rho^a + v_2\,\rho^b \\ v_0\,\rho^c + v_2\,\rho^a \\ v_0\,\rho^b + v_1\,\rho^c \\ v_0\,\rho^a + v_1\,\rho^b + v_2\,\rho^c \end{pmatrix} = \begin{pmatrix} u_0 \\ u_1 \\ u_2 \\ u_3 \end{pmatrix}$$

and we need to prove that at least one of the u_i, $0 \leq i < 4$, must be non-zero. Assuming the contrary, we obtain $v_2 = v_1\,\rho^{a-b}$ from $u_0 = 0$ and $v_0 = v_1\,\rho^{-b+c}$ from $u_2 = 0$. Substituting in the relation for $u_3 = 0$ we obtain

$$0 = v_1\,\rho^{a-b+c} + v_1\,\rho^b + v_1\,\rho^{a-b+c} = v_1\,\rho^b$$

whence $v_1 = 0$, a contradiction.

Note that the branch number is tight. For vectors V of weight between one and three and entries v_i which are either 0 or $1 + \rho + \rho^2 + \ldots + \rho^{m-1}$, it is $\mathsf{wt}(V) + \mathsf{wt}(U) = 4$. □

Remark 1. If $m = 4$ or 8, then *all* matrices (2) are invertible. In fact, for $m = 4$ the determinant $\rho^{4a} + \rho^{4b} + \rho^{4c}$ is always 1 and for $m = 8$ it is equal to either 1 or ρ^4.

Now let us consider the matrices of type (2) with equally lightweight inverse.

Theorem 2. *Let $R_m = \mathbb{F}_2[X]/(X^m + 1) = \mathbb{F}_2[\rho]$ be defined as in Theorem 1. The Almost MDS matrices $M = \mathrm{circ}(0, \rho^a, \rho^b, \rho^c)$ that admit an inverse of the same form $\overline{M} = \mathrm{circ}(0, \rho^d, \rho^e, \rho^f)$, i.e. with entries of weight at most one, are those that satisfy $a \equiv c + \tau$ where $2\,\tau \equiv 0 \bmod m$ (for odd m this implies $\tau = 0$, for even m it can be $\tau = 0$ or $m/2$). In this case, the parameters of the matrix $\overline{M}$ are: $d \equiv a - 2\,b$, $e \equiv -b$, and $f \equiv d + \tau \bmod m$.*

The involutory matrices M are those for which, additionally, $2\,b \equiv 0$.

Proof. Since M and $\overline{M}$ are circulant, $M \cdot \overline{M} = I$ is equivalent to

$$\begin{pmatrix} \rho^{a+f} + \rho^{b+e} + \rho^{c+d} \\ \rho^{b+f} + \rho^{c+e} \\ \rho^{a+d} + \rho^{c+f} \\ \rho^{a+e} + \rho^{b+d} \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \end{pmatrix} \tag{3}$$

Ignoring for the moment the first component in (3), we see that

$$\text{(i)}\ \ b + f \equiv c + e\ \ , \quad \text{(ii)}\ \ a + d \equiv c + f\ \ , \quad \text{(iii)}\ \ a + e \equiv b + d \bmod m\ \ .$$

Upon adding relations (i), (ii) and (iii) we get $2\,a + (d + e + f) = 2\,c + (d + e + f) \equiv 0$, in other words $2\,(a - c) \equiv 0 \bmod m$, whence $a \equiv c + \tau$ where $2\,\tau \equiv 0 \bmod m$. Replacing this into (ii) we obtain at once $d \equiv f + \tau$, which is equivalent to $f \equiv d + \tau$. In particular, $a + f \equiv c + d \bmod m$ holds always, which means that ρ^{a+f} and ρ^{c+d} cancel out in the first relation in (3), and it must be $\rho^{b+e} = 1$, i.e. $e \equiv -b \bmod m$. Finally, substituting in relation (iii) we obtain $d \equiv a - 2\,b \bmod m$.

The last statement is proved by substituting $a \equiv d$ into $d \equiv a - 2\,b \bmod m$. □

Remark 2. Hence, there are two degrees of freedom to define the matrices M, and one extra bit in case m is even. If M has to be involutory, there is only one degree of freedom (and two bits for even m).

Remark 3. An analogue of Theorem 1 holds over fields $\mathbb{F}_{2^m}$ as well. A matrix $M = \mathrm{circ}(0, A, B, C) =$ with $ABC \neq 0$ is invertible if and only if its determinant $A^4 + B^4 + C^4 \neq 0$, in which case it is Almost MDS. However, the multiplications by A, B, or C are circuits of non-negligible depth, unless they are equal to 1.

Remark 4. This is not the only work on diffusion matrices on rings other than a finite field. Dehnavi et al. [DRS+14, DRS15, RDS+14] study matrices over rings $\mathbb{F}_2^m$ where the ring operations are the bitwise XOR and AND. They build MDS matrices whose entries are sums of rotations or shifts. Their research does not consider Almost MDS matrices, and even though their operations are efficient, they are not concerned with minimising the depth. In fact, their diffusion layers have a much higher latency than ours. Also for the MDS matrices in [SS17] the goal is to minimise the total XOR count and their matrices lead to circuits with longer critical path.

3.1.2 Selection

Besides the MIDORI circulant $M_0 := \mathrm{circ}(0, 1, 1, 1)$ we considered several other matrices.

1. For QARMA-64 ($m = 4$) we initially restricted our attention to the involutory matrices

$$\begin{aligned} M_{4,1} &= Q_{4,1} = \mathrm{circ}(0, \rho, \rho^2, \rho^3)\ \ , \\ M_{4,2} &= Q_{4,2} = \mathrm{circ}(0, \rho, \rho^2, \rho)\ \ , \text{ and} \\ M_{4,3} &= Q_{4,3} = \mathrm{circ}(0, 1, \rho^2, 1)\ \ . \end{aligned}$$

2. For QARMA-128 ($m = 8$) we considered

$$M_{8,1} = Q_{8,1} = \mathrm{circ}(0, \rho^2, \rho^4, \rho^6) \ ,$$
$$M_{8,2} = Q_{8,2} = \mathrm{circ}(0, \rho, \rho^4, \rho^5) \ ,$$
$$M_{8,3} = Q_{8,3} = \mathrm{circ}(0, 1, \rho^4, 1) \ , \text{ and}$$
$$M_{8,4} = \mathrm{circ}(0, \rho, \rho^2, \rho^5) \quad \text{with} \quad Q_{8,4} = \mathrm{circ}(0, \rho, \rho^4, \rho) \ .$$

Note that $M_{8,1}$, $M_{8,2}$, $M_{8,3}$, and $Q_{8,4}$ are involutory, and $\overline{M}_{8,4} = \mathrm{circ}(0, \rho^5, \rho^6, \rho)$.

We selected these matrices and then among them according to various criteria, which we describe in the following. (The order does not signify importance.)

The first criterion is the number of fixed points of the matrix Q, in order to improve the cryptographic properties of the central construction (cf. § 4.4). From this points of view, optimal involutory matrices over R_4 are $Q_{4,2}$ and $Q_{4,3}$, that have the optimal number of 2^{32} fixed points (cf. Lemma 1 in [SBY+15]), and $Q_{8,4}$ that attains the minimum of 2^{64} fixed points. Note that $Q_{4,1}$ has 2^{48} fixed points, $Q_{8,1}$ has 2^{96} fixed points, and $M_{8,2} = Q_{8,2}$ has 2^{72} fixed points (cf. Appendix C), which is close to optimal.

The second criterion is the number of active S-Boxes in linear and related-tweak differential trails. In [MWGP11] it is shown how to use mixed integer linear programming (MILP) to count the active S-Boxes in these trails. The technique has been extended to tweakable ciphers in [BJK+16]. An important part of the MILP method is the determination of a set of equations to model the action of the diffusion matrix M, i.e. the set of possible combinations of active cells in a state column and in the column obtained by multiplying it by M. Different Almost MDS matrices do not necessarily have the same diffusion patterns, even if we restrict to circulants of the form $\mathrm{circ}(0, x, y, z)$ with non-zero x, y, and z. Hence, we have first determined the diffusion patterns for each matrix, by means of a simple and fast exhaustive enumeration over all possible inputs. Our matrices fall into two different classes:

1. *Class I* includes M_0, $M_{4,1}$ and $M_{8,1}$; and
2. *Class II* includes $M_{4,2}$, $M_{4,3}$, $M_{8,2}$, $M_{8,3}$, $M_{8,4}$ and $Q_{8,4}$.

Other matrices with less relative differences in the rotations or, if involutory, with more fixed points, have not been taken into account.

Class I matrices have 51 possible active column-to-column state transitions, of which the 67 transitions of Class II matrices are a superset. These transitions are depicted in Figures 4 and 5, for Class I and II matrices, respectively. We note that the exhaustive enumeration method yields the following result.

Proposition 1. *All the matrices over R_4 and R_8 characterised in Theorem 2 (i.e. the circulants $M = \mathrm{circ}(0, \rho^a, \rho^b, \rho^c)$ that admit an inverse of the same form) have only two possible column-to-column active state transition sets, namely those in Figures 4 and 5.*

We modelled these two different behaviours in MILP to compute lower bounds for the numbers of active S-Boxes in both the linear (and differential) and related-tweak settings. Note that in the related-tweak model we are not including the key. We display these results in Table 3. The results for the linear setting have been computed on the whole cipher for $r \leq 6$ and just half of the cipher (only forward rounds) for $r \geq 7$, in which case the objective value has been doubled. In the related-tweak setting, for $r \geq 9$ we either model only half of the cipher or use a dual bound for the whole cipher.

The structures of QARMA and MANTIS are not identical, hence it is possible that a different tweak permutation could give better bounds for QARMA. However, the bounds for

Figure 4: The Column-wise Active State Transitions for Class I Matrices

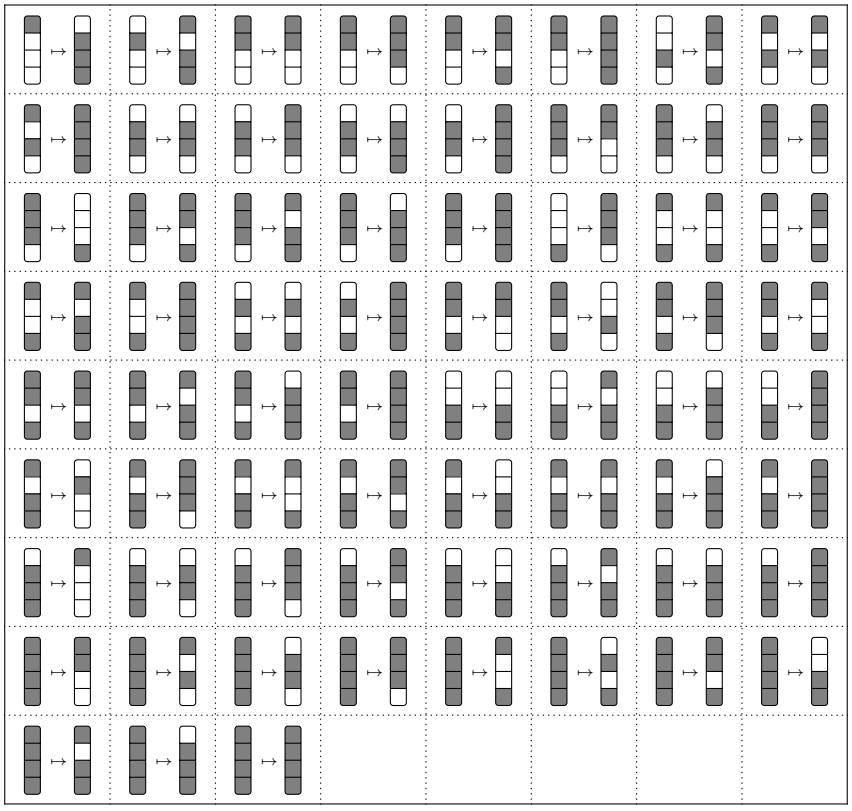

Figure 5: The Column-wise Active State Transitions for Class II Matrices

Table 3: Lower Bounds on the Number of Active S-Boxes

QARMA with Class I Diffusion Matrices

r	3	4	5	6	7	8	9	10	11
Linear	32	50	64	≥ 70	≥ 76	≥ 82	≥ 100	≥ 114	≥ 124
Rel. Tweak	16	24	34	44	52	60	≥ 64	≥ 70	≥ 78

QARMA with Class II Diffusion Matrices

r	3	4	5	6	7	8	9	10	11
Linear	32	50	60	≥ 68	≥ 74	≥ 82	≥ 98	≥ 112	≥ 120
Rel. Tweak	14	24	30	42	48	58	≥ 62	≥ 68	≥ 76

Comparison: MANTIS (Class I Diffusion Matrix and no ShuffleCells in Reflector)

r	3	4	5	6	7	8
Linear	32	46	60	≥ 68	≥ 76	≥ 82
Rel. Tweak	12	20	34	44	50	56

the two ciphers are quite close (and sometimes slightly better because of the improved diffusion in the center, cf. § 4.4), so we decided to keep the same tweak permutation.

The bounds on the active S-Boxes between the two matrix classes are also very similar. Hence, the class of a matrix will not be used to choose one matrix over another unless all other parameters are equal.

The third criterion is the overall minimisation of the maximum probability of non-trivial differentials on two active cells inside a column through the following operations: an S-Box layer, the diffusion matrix and a second S-Box layer. For each input differential (two cells) we determine the output differential (also on two cells) with the highest likelihood, and then we average these probabilities. The minimum of this average is achieved with $M_{4,2}$ and $M_{8,4}$ for all chosen S-Boxes (see §§ 3.3 and 3.4).

For $m = 4$ and with the S-Box σ_1, the average occurrence of a differential goes from 31.64/256 with the MIDORI circulant, to 18.11/256 with $M_{4,3}$, then 16.94/256 with $M_{4,1}$, and finally to 8.52/256 with $M_{4,2}$. (Actually, a slightly better value, 8.28/256 could be reached with $M' = \mathrm{circ}(0, 1, \rho, 1)$, but this matrix is non-involutory, and therefore the fourth criterion below would not apply.)

For $m = 8$ and with the 8-bit S-Box constructed from σ_1 (cf. § 3.4), the average goes from $\approx 253/2^{16}$ with M_0 and $M_{8,3}$ to $\approx 140/2^{16}$ with $M_{8,1}$ to values $\approx 28/2^{16}$ with $M_{8,2}$, $M_{8,4}$ or $Q_{8,4}$.

Changing S-Boxes does not significantly alter these values.

This proves that the desired goal of increased independence of the outputs, and corresponding disruption of differential characteristics, can be achieved.

Finally, the fourth criterion is the applicability of our analysis of invariant subspace attacks (cf. § 4.3) and its outcome. The analysis assumes a single involutory matrix is used in order to get invariant subspaces that are as close as possible to the whole space. This prompted us to choose $M_{8,2}$ over the pair ($M_{8,4}$, $Q_{8,4}$) that was used in a previous revision of QARMA-128. Using $M_{8,2}$ for the whole cipher seems to provide better resistance against these attacks than using the other shortlisted matrices.

In light of these observations, *we choose the matrix $M_{4,2} = Q_{4,2}$ for QARMA-64, and the matrix $M_{8,2} = Q_{8,2}$ for QARMA-128.*

3.2 The Central Construction

Reflection attacks are a specific class of cryptanalysis originally developed against 2K-DES [Kar07] but that can be particularly efficient against PRINCE-like ciphers [SBY+15]: If the reflector has too many fixed points, there will be a self-differential with a zero difference after the round key addition at its sides, that may propagate outwards with a palindromic structure if a matching self-differential with a difference of α exists. With respect to MANTIS, by design we took an important step towards resistance against such attacks. The matrix we chose for QARMA-64 has 2^{32} fixed points out of 2^{64} values, which is the minimum for a linear function (cf. Lemma 1 in [SBY+15]), whereas the MIDORI and MANTIS circulant M_0 has 2^{48} (the matrix for QARMA-128 has 2^{72} fixed points out of 2^{128}, close to the minimum of 2^{64}, whereas M_0 has 2^{96} fixed points over $R_8^{4\times4}$). But, we decided to take also an additional countermeasure.

Suitable whitening around the reflector can prevent these attacks, provided that the two whitenings always have a non constant difference as a function of the whitening key. The PRINCE whitening key expansion was designed this way [BCG+12, § 3.4]: for any value of z, equation $w^0 + w^1 = z$ has exactly one solution. In fact, the map $w^0 \mapsto o(w^0) = w^1$ is a (linear) orthomorphism, i.e. *both* o and $x \mapsto x + o(x)$ are *permutations*. Thus, an attacker that does not know w^0 cannot make any assumption about the difference after the whitening. Since the same tweak is added on both sides, it does not affect the difference.

The central addition of a core key derived round key serves to continue the regular key mixing in the cipher and also to make the pseudo-reflector in general not involutory, with an unpredictable difference at its sides. Since there are values of the core key for which the central rounds are involutory, we intentionally do not add also the tweak in the middle to prevent attacks that could exploit this.

An analysis of the security implications of our central construction is given in § 4.4.

Finally, the cryptanalysis in [Din15], even though it does not (yet?) lead to practical attacks, further motivates our departure from pure FX-constructions, especially since QARMA's permutations are not ideal.

3.3 Selection of the 4-Bit S-Boxes

Despite the existence of excellent classifications of 4-bit S-Boxes [LP07, Saa11], choosing the right S-Box for a given application is still a non-trivial task. The cipher's designer must balance on one side desirable cryptographic properties and on the other one performance and cost aspects, such as critical path length and area. The critical path of a circuit is the longest sequence of sequential operations from an input to an output: As it determines the latency of the circuit, for our applications it is a more important parameter than area.

In the case of PRINCE the S-Box was chosen to be 4-bit to reduce total complexity and critical path, but once this decision had been made, the choice seems to have been driven essentially by cryptographic properties. At the opposite side of the spectrum, the MIDORI designers did not hesitate to choose an S-Box with cryptographic properties that are not optimal. Indeed, the particular values of the four fixed points and the limited round key mixing allowed a vast class of weak keys to be found [GJN+17]. The problem here were not the fixed points per se, but the fact that the set of their values were preserved under round key addition and linear layer.

QARMA can use three S-Boxes: a very small involutory one with complexity similar to that of MIDORI, but with improved cryptographic properties; an S-Box which is affine equivalent to the S-Box S_6 defined in the PRINCE paper; and an involutory one designed so that its complexity falls in between.

A key instrument in choosing these S-Boxes is an automated approach for determining S-Boxes with bounded path delay. The designers of MIDORI observe that *the path delay is highly related to the dependency of the computation.* The path delay is estimated using

a metric called the *depth* of a circuit, defined as *the sum of sequential path delays of basic operations such as* AND*,* OR*,* NAND*,* NOR*,* XOR*,* XNOR *and* NOT *(we also consider ternary and quarternary operators).*

To find the MIDORI S-Boxes, all involutory S-Boxes have been examined in order of increasing depth. This method can be slow and memory intensive, for instance, if all S-Boxes have to be generated first, synthesized and then sorted. We replace these expensive search methodologies by slightly less precise heuristics.

3.3.1 Identifying S-Boxes with Almost Optimal Path Delay

Let us consider the MIDORI S-Box Sb_0 as an example. The SOP (sum-of-products) of the least significant bit of the output corresponds to the Boolean function $wy\overline{z}+w\overline{y}z+x\overline{z}+x\overline{y}$ on the variables $\{w, x, y, z\}$ ($\overline{x} := \mathsf{NOT}(x)$). Applying de Morgan's theorem, we see that this can be evaluated as the NAND, actually the NAND4 of the four expressions $\mathsf{NAND3}(w, y, \overline{z})$, $\mathsf{NAND3}(w, \overline{y}, z)$, $\mathsf{NAND2}(x, \overline{z})$, and $\mathsf{NAND2}(x, \overline{y})$. A NAND4 gate can be implemented (with some complication because of wiring) with a depth of 2, and thus by using a layer of NOT's of depth 0.5, a layer with two NAND3's and two NAND2's with a depth of 1.5, and the NAND4 gate, this output bit can be implemented with a depth of 4. However, the SOP of the *negation* of the same output bit is $\overline{x}\overline{y}\overline{z} + \overline{w}\overline{x} + yz$ which can be implemented as the NAND3 of $\mathsf{NAND3}(\overline{x}, \overline{y}, \overline{z})$, $\mathsf{NAND2}(\overline{w}, \overline{x})$, and $\mathsf{NAND2}(y, z)$ – with a total depth of 3.5. Negating it and applying de Morgan's theorem again, the original function can be represented as the NOR3 of $\mathsf{NOR3}(x, y, z)$, $\mathsf{NOR2}(w, x)$, $\mathsf{NOR2}(\overline{y}, \overline{z})$, which has a depth of 3. Similar considerations hold for the other output bits.

Remark 5. In general, considering the SOP and the NOT-SOP of each output bit we can always match the claimed minimal depth of all the S-Boxes from the literature that we have sampled, or exceed this depth by at most 0.5.

Therefore we sieve the S-Boxes by computing the SOP and NOT-SOP of each output bit by the Quine-McCluskey algorithm [Qui52, McC56]. We initially consider, for instance, only those S-Boxes whose output bits can be all expressed as sums of at most three products, each one having at most weight 3 – or as the negations of products of at most three sums of weight at most 3. We verify if these S-Boxes satisfy additional cryptographic constraints – and if no matching S-Boxes are found we gradually increase the complexity of the allowed expressions. By allowing generalised NAND and NOR gates, we get tighter upper bounds for the depth. Gradually, we restrict the cryptographic properties until we get a manageable set of S-Boxes that can be further studied.

3.3.2 Desirable Cryptographic Properties

For various ciphers, there are several interesting cryptographic properties that have been required for S-Boxes, depending on the requirements of the cipher and the structure of other cipher components. For 4-bit S-Boxes, optimal properties are the following:

S1 *The maximal probability of a differential is* $1/4$.

S2 *The maximal absolute bias of a linear approximation is* $1/4$.

S3 *Each of the 15 non-zero component functions has algebraic degree* 3.

Bijective 4-Bit S-Boxes that satisfy Properties **S1** and **S2** are indeed called *optimal* in [LP07], where they are shown to satisfy additional properties, such as optimal resistance against algebraic cryptanalysis, a fact which we shall exploit in the security analysis. Sometimes the first two properties are strengthened, as in the PRINCE S-Boxes, as follows:

S1' *Property **S1** holds and there are exactly 15 differentials with probability* $1/4$.

Table 4: Summary of Properties of the MIDORI, PRINCE and QARMA S-Boxes

S-Box	MIDORI	PRINCE		σ_0	σ_1	σ_2	
		Direct	Inverse			Direct	Inverse
Max. prob. of a differential	1/4	1/4	1/4	1/4	1/4	1/4	1/4
# with max. probability	24	15	15	18	15	15	15
Max. bias of a lin. approx.	1/4	1/4	1/4	1/4	1/4	1/4	1/4
# with max. bias	36	30	30	32	30	30	30
Algebraic Degree	3	3	3	3	3	3	3
# components of deg 3, 2	12, 3	15, 0	15, 0	14, 1	15, 0	15, 0	15, 0
Algebraic Immunity	2	2	2	2	2	2	2
Branch Number	2	2	2	2	2	2	2
Fixed Points	4	0	0	2	0	0	0
Minimal depth (GE)	3.5	5	4.5	3.5	4	4.5	4
Minimal area (GE)	12.8	20.2	19	14.17	16.5	20.2	19

S2' *Property **S2** holds and there are exactly 30 linear approximations with absolute bias* 1/4.

Finally, another important property is the *full diffusion* property:

S4 *Each input bit of the S-Box shall influence each output bit non-linearly.*

This property is easily verified from the SOP of each output bit: each input bit should be present in each output bit in at least one product of weight at least two.

Property **S4** is very important in the design of the second MIDORI S-Box, to ensure proper diffusion of the input bits to all bits of the state. We shall consider also of the following slightly relaxed version of Property **S4**, that, as defined in § 3.4, will suffice to ensure an analogous diffusion property with our design:

S4' *At least three input bits of the S-Box influence each output bit non-linearly, and there is at most one input bit that influences only three output bits non-linearly;*

We now describe the three S-Boxes that we have selected for QARMA and compare them to the MIDORI and PRINCE S-Boxes, whose properties have been the starting point for our own investigation. Several properties of these S-Boxes are summarised in Tables 4 and 5. Minimal depth and minimal area refer to our implementation (cf. Section 5).

3.3.3 A Lightweight MIDORI-like S-Box with Improved Cryptographic Properties

The MIDORI S-Box has four fixed points, three non-zero component functions of algebraic degree just 2, and it satisfies Properties **S1**, **S2**, and **S4'**, but not **S1'** and **S2'**. We aim at improving on some of these parameters.

We use the candidate sieving technique described at the end of § 3.3.1 to search the involutory S-Boxes with no more than four fixed points. In order to search them all, we modify Prissette's algorithm [Pri10] to enumerate all fixed-point free involutions over a specific set to generate all involutions with a specific subset of fixed points.

We found an alternative with the same depth but only two fixed points, and only one non-zero component function of algebraic degree 2 instead of three.

This S-Box, that we propose for the lightest versions of QARMA is

$$\sigma_0 := [\,0, 14, 2, 10, 9, 15, 8, 11, 6, 4, 3, 7, 13, 12, 1, 5\,] \ .$$

It has two fixed points instead of four. Interestingly, this S-Box improves on the MIDORI one also on the number of differentials and linear approximations of maximal likelihood or

Table 5: The Bit-Flipping Pattern of Various S-Boxes

σ_0 & MIDORI	Times b bits are flipped			
Flipping bit	one	two	three	four
0	5	2	1	0
1	4	4	0	0
2	5	2	1	0
3	2	4	2	0

σ_1	Times b bits are flipped			
Flipping bit	one	two	three	four
0	3	4	1	0
1	3	4	1	0
2	3	4	1	0
3	3	4	1	0

σ_2	Times b bits are flipped			
Flipping bit	one	two	three	four
0	2	1	4	1
1	2	4	2	0
2	2	3	2	1
3	6	2	0	0

$\overline{\sigma}_2$	Times b bits are flipped			
Flipping bit	one	two	three	four
0	5	1	1	1
1	2	3	2	1
2	2	5	0	1
3	3	2	3	0

PRINCE	Times b bits are flipped			
Flipping bit	one	two	three	four
0	5	2	1	0
1	2	2	4	0
2	4	4	0	1
3	4	2	2	0

PRINCE inv	Times b bits are flipped			
Flipping bit	one	two	three	four
0	4	2	2	0
1	2	3	2	1
2	4	1	2	1
3	5	2	1	0

bias: respectively 18 and 32 versus 24 and 36 for the MIDORI S-Box (cf. Table 4). Hence it satisfies Properties **S1**, **S2**, and **S4'**, and with respect to Properties **S1'** and **S2'**, even though it does not satisfy them, it is closer to the ideal minima than the MIDORI S-Box.

We also consider the output weight distribution of single bit input differentials. For our S-Boxes we display this in Table 5. The MIDORI S-Box and σ_0 have the same the bit-flipping patterns, and a single bit input difference will produce a single bit output difference with likelihood 1/2.

The critical path is 3.5 GE as for the MIDORI S-Box, its area is 14.17 GE in the process used in our implementation (compared to 12.8 GE for the MIDORI S-Box), cf. Section 5.

The search lasted only a few seconds on a single core of a 2,4 GHz Intel Core i7 laptop.

Interestingly, we found no S-Box where the values of the two fixed points differ on more than one bit, which implies that in this situation it is important *that the diffusion layer does not always map these values onto themselves and the round constants influence other bits as well, to prevent characteristics that map a small subset of possible states onto itself.* This is one of the motivations in the design of the linear layer in § 3.1.

3.3.4 A Compact, Optimal, Involutory S-Box

We found also the following involutory S-Box with no fixed points:

$$\sigma_1 := [\,10, 13, 14, 6, 15, 7, 3, 5, 9, 8, 0, 12, 11, 1, 2, 4\,] \ .$$

The output weight distribution for single bit input differentials is given in Table 5. For any b, each input bits has the same number of 1-bit to b-bit differentials – this is the reason we call this S-Box *homogeneous*. This may make individual bits less distinguishable from each other, for instance under a threat model for physical attacks that takes into account power consumption. σ_1 satisfies Properties **S1'**, **S2'**, **S3** and **S4**. Its depth is 4 and the area is 16.5 GE. This is the default S-Box for QARMA in all its versions.

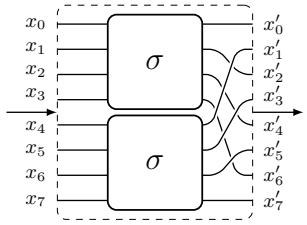

Figure 6: The Construction of the 8-bit S-Box Σ of QARMA-128

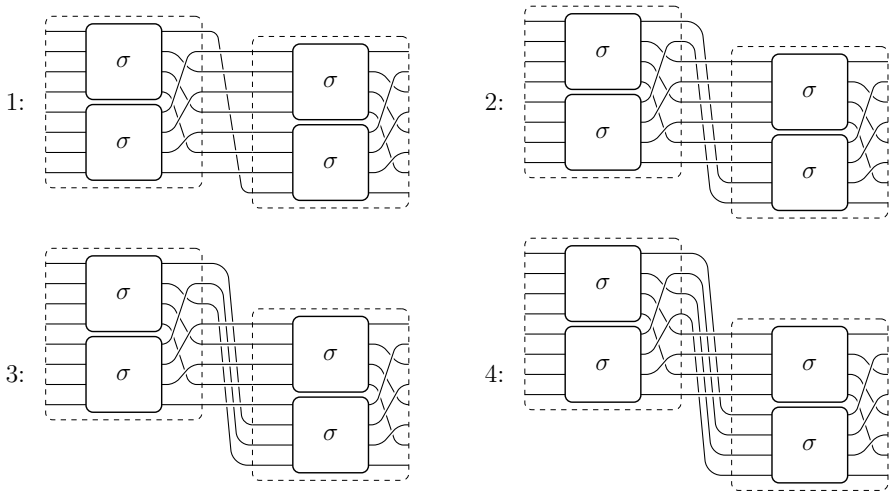

Figure 7: Alignment of Output and Input Bits of Consecutive Instances of the 8-bit Composite S-Box

3.3.5 The Lightweight S-Box from the PRINCE Family

It may be requested that the S-Box is non-involutory. For this purpose, we have filtered all the S-Boxes allowed for PRINCE, which fall into eight affine equivalence classes. These S-Boxes all satisfy Properties **S1'** and **S2'**, as well as **S3** and **S4**.

In fact, we selected an S-Box which is affine equivalent to the S-Box S_6 as defined in the PRINCE paper, because it and its inverse can be implemented with depths of 4.5 and 4 respectively, which seem to be optimal among all PRINCE S-Boxes according to the bounds provided by our search tools. This S-Box and its inverse are:

$$\sigma_2 = [\,11, 6, 8, 15, 12, 0, 9, 14, 3, 7, 4, 5, 13, 2, 1, 10\,]$$
$$\overline{\sigma}_2 = [\,5, 14, 13, 8, 10, 11, 1, 9, 2, 6, 15, 0, 4, 12, 7, 3\,]\ .$$

These depths are smaller than those required by the default PRINCE S-Box, and the area is the same – approximately 20.2 GE for the S-Box and 19 GE for its inverse.

All the non-zero component functions have algebraic degree 3. The S-Box has no fixed points and the output weight distribution for single bit input differentials for both σ_2 and $\overline{\sigma}_2$ is given in Table 5. It has are less single bit to single bit differentials than the standard PRINCE direct and inverse S-Boxes.

3.4 The 8-bit S-Boxes

As in MIDORI-128 we construct an 8-bit S-Box Σ by placing two instances of a single 4-bit S-Box in parallel. However, we wire the input and output bits in a single and simpler

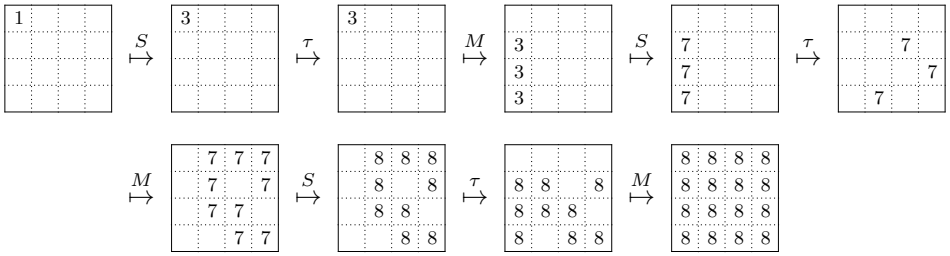

Figure 8: Three-Round Full Diffusion Property

way, as shown in Figure 6, which is asymmetric. The S-Box σ is one of the S-Boxes σ_i described in § 3.3. If we write a 8-bit cell of the state as $(x_7, x_6, x_5, x_4, x_3, x_2, x_1, x_0)$, σ is applied to (x_7, x_6, x_5, x_4) producing the output bits (x'_7, x'_5, x'_3, x'_1), and to (x_3, x_2, x_1, x_0) producing the output bits (x'_6, x'_4, x'_2, x'_0), and the output of the combined 8-bit S-Box is $(x'_7, x'_6, x'_5, x'_4, x'_3, x'_2, x'_1, x'_0)$. Since the construction is not symmetric, the opposite wiring must be implemented for $\overline{\Sigma}$.

The rationale behind this is that in combination with the chosen matrices M, if the output of Σ is cyclically rotated by any amount of bits and then fed into another instance of Σ, exactly 2 bits of the output of one of the two 4-bit S-Boxes in the first Σ are wired into each of the two 4-bit S-Boxes of the second instance of Σ. This is clear because all the outputs of one instance of σ will be wired into the even numbered bits of the output of Σ, and the outputs of the other instance of σ into the bits in the odd numbered positions of the output of Σ, so any group of four cyclically adjacent output bits of Σ will be equally partitioned among the two sources. This is also graphically depicted in Figure 7.

If the 4-bit S-Box σ satisfies Property **S4**, a *three-round full diffusion property* as in MIDORI-128 (Theorem 1 in [BBI+15]) still holds, namely *any input bit nonlinearly affects all 128 bits of the state after 3 full rounds* (i.e. not short rounds). This serves in order to avoid having an independence property similar to the one exploited in the attack on full-round KLEIN [LN14], in other words to make sure that the cipher does not act like two 64-bit ciphers side by side. Full diffusion holds even under weaker Property **S4'**, which means that the S-Box σ_0 can be used also in QARMA-128. In the following Theorem we give a proof of this fact.

Theorem 3. *In QARMA-128, any input bit nonlinearly affects all 128 bits of the state after three rounds, intended as the first short round, two full rounds, and the diffusion layer of the following one.*

Proof. Let us assume the weaker Property **S4'**, and use Figure 8 as a graphical aid.

An input bit affects (at least) three bits in the same 8-bit cell after the first S-Box layer. Because of the construction of the 8-Bit S-Box these bits are all in either the even-numbered or the odd-numbered output bits. Hence, regardless of the rotation induced by the matrix layer, two bits influence one half of the inputs to a 8-bit S-Box in the next round, and at least one bit influences the other half of the inputs to that S-Box.

This means that after the second S-Box layer at least $7 = 4 + 3$ bits are affected per cell, which implies that all 8 output bits are affected after the third S-Box layer. □

3.5 The ω Function

The indexes of the tweak register modified by ω have been chosen along the length 14 cycle of the tweak cell permutation - and one along the length 2 cycle $(1, 5)$ - in order to minimise the number of cell values with the same difference to the corresponding cells

of the round constants (and so make partial slide attacks less likely) and maximise the spread of different values if one starts with a non-zero tweak with all equal cells.

3.6 The Round Constants

We did not consider sparse constants: since they are hardwired, and XOR and XNOR have the same costs, generating the round tweakey has the same cost regardless of the Hamming weight of the constants.

4 Security Analysis

Security Model. *We shall assume the attacker does not have control on the key, but she may have full control on the tweak. A TBC is understood to offer n bits of (time-data tradeoff) security if no better attacks are possible than time $2^{n-d-\epsilon}$ with 2^d chosen or known {plaintext, ciphertext, tweak} triples, for a small ϵ (e.g. 2).*

Remark 6. In some applications, such as memory encryption, adversarial control on the tweak may be somewhat limited, for instance it may be only passive. A further technique to reduce adversarial control on the tweak used in the block cipher itself has been introduced in § 2.9. Designing a cipher under the above security model will therefore exceed the requirement of such a use case, strengthening the security margins.

4.1 Common Attacks on Block Ciphers

One of our design goals was to be able to carry over as much of the security analysis of MIDORI and MANTIS as possible while at the same time improving both diffusion characteristics and the bounds deriving from the cryptanalysis of Even-Mansour schemes. This is achieved by giving the forward rounds a similar structure to the MIDORI rounds.

Linear and Differential Cryptanalysis. Looking at Table 3, we see that no related-tweak linear or differential distinguisher based on a characteristic has likelihood better than 2^{-60} for QARMA-64 already when $r = 5$, and note that not only the bounds are not tight (bounds would be only tight if the S-Boxes could be chosen freely for each cell and round), but they also express a security level without taking whitening into account. (With $r = 6$ this probability already decreases to 2^{-88}.) QARMA_7-64 has four more rounds, which should provide a sufficient security margin.

QARMA-128's state can be viewed as thirty-two 4-bit cells, but we did not find this partition useful for a security analysis based on trails. The main reason is that the 8-bit S-Box construction Σ (cf. Figure 6), has branch number 2 viewed as a map from two 4-bit cells (x_7, x_6, x_5, x_4) and (x_3, x_2, x_1, x_0) to two 4-bit cells $(x'_{i+7}, x'_{i+6}, x'_{i+5}, x'_{i+4})$ and $(x'_{i+3}, x'_{i+2}, x'_{i+1}, x'_{i+0})$ for any value of i with the indexes taken mod 8. Hence, if the input is active, only one of the two 4-bit outputs is guaranteed to be active. Heuristically, after the first round we can estimate that if one of the two 4-bit outputs in a 8-bit cell is active, then the other output is active with a likelihood of 75%. This tells us that the actual active S-Box counts are most likely much larger than the counts given in Table 3, and for this reason we did not require $r = 12$ to assume security against linear and differential attacks for QARMA-128. However, a bound along these lines can probably only be proved by converting the cell-wise MILP model into a bit-wise one, which would be computationally infeasible, except for a very small number of rounds.

Slide Attacks. The cipher uses four different types of rounds, five if we include the pseudo-reflector. Even similar rounds are distinguished by the addition of the different round constants. Hence, it seems that slide attacks can not present a threat to QARMA.

Table 6: Upper Bounds on the Algebraic Degree of QARMA

Rounds	1	2	3	4	5	6	7	8	···
QARMA-64	3	9	27	51	59	62	63	63	···
QARMA-128	3	9	27	81	112	122	126	127	···

Impossible Differential and Zero-Correlation Linear Cryptanalysis. To verify the resistance against these attacks, we use the characteristic matrix technique from [SLG+16]. It is easy to verify that either going through three rounds or (at least) two rounds and the pseudo-reflector one obtains a all-non-zeros product of characteristic matrices, concluding that $r = 3$ is already sufficient to avoid the attacks.

Boomerang, Integral and Meet-in-the-Middle Attacks. The arguments in the MIDORI paper against these attacks can be applied straightforwardly. For instance, it should be difficult to construct meet-in-the-middle attacks on 11 rounds on QARMA-64 or QARMA-128. Indeed, there is a meet-in-the-middle key recovery attack to 10-round QARMA-64 with the outer whitening removed [ZD16] that uses 2^{53} chosen plaintexts and has a time complexity of $2^{70.1}$ encryptions, with a memory footprint of $2^{123.6}$ bits. Against 10-round QARMA-128 without outer whitening, the same attack has complexities of 2^{105} chosen plaintexts, time $2^{141.7}$ encryptions, and $2^{240.6}$ bits of memory. It does not seem to be extendable further.

4.2 Algebraic Attacks

We consider the applicability of algebraic attacks [CP02] on $\texttt{QARMA}_r$-n. For this purpose we view the S-Box layer of QARMA-128 as 32 4-bit S-Boxes and consider the output bit intertwining as part of the diffusion layer.

We first argue that QARMA-n has sufficiently many rounds to reach maximum algebraic degree $n-1$. Using [BCC11, Theorem 2], we see that if F_i is the function formed by the first i rounds of QARMA-n, the following bound

$$\deg(F_{i+1}) \leq \min\left(3 \cdot F_i,\ n - \left\lceil \frac{n - \deg(F_i)}{\gamma} \right\rceil\right)$$

holds, where for our (4-bit) S-Boxes $\gamma = 3$. We thus obtain the upper bound for the algebraic degree in Table 6, from which we infer that QARMA-64 and QARMA-128, that have at least 12 and 18 rounds in their most aggressive versions, should have sufficiently many rounds to reach maximum degree.

Let us count the quadratic equations and variables. Our 4-bit S-Boxes are optimal according to the definition in [LP07], where it is also proved that they are described by $e = 21$ quadratic equations in the $v = 8$ input/output variables over $\mathbb{F}_2$. Hence, the entire system for a fixed-key $\texttt{QARMA}_r$-n permutation consists of $\frac{n}{4} \cdot (2r+2) \cdot e$ quadratic equations in $\frac{n}{4} \cdot (2r+2) \cdot v$ variables. For $\texttt{QARMA}_7$-64 this translates to 5376 equations in 2048 variables, and for $\texttt{QARMA}_{11}$-128 to 16128 equations in 6144 variables. For comparison, a fixed-key AES-128 permutation consists of 6400 equations in 2560 variables, and a fixed-key AES-256 permutation consists of 8960 equations in 3584 variables.

While it is still unclear whether algebraic attacks can be effective against the AES, it is also clear that QARMA-64 may be more amenable to such attacks. However, the sizes of the systems are roughly of the same magnitude, suggesting similar resistance. The situation is akin to that of PRINCE and MANTIS, whose equation systems are slightly smaller. The situation is very different for QARMA-128, where the system is significantly larger than AES-128's – and in fact even larger than the system for AES-256 – implying that QARMA-128 should offer much better resistance against algebraic cryptanalysis.

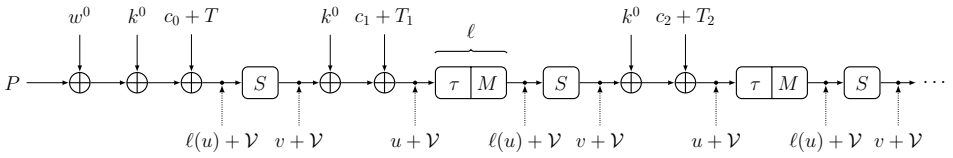

Figure 9: Propagation of Invariant Subspace Cosets in the Forward Rounds

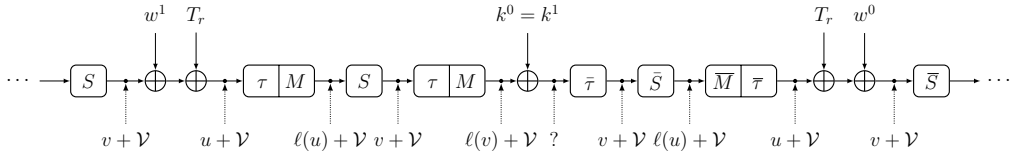

Figure 10: Propagation of Invariant Subspace Cosets Through the Central Construction

4.3 Invariant Subspace Cryptanalysis

Invariant subspace attacks such as those successfully mounted against `MIDORI`-64 [GJN+17] are a threat to any key-alternating block cipher with a `PRINCE`-like key schedule, including `QARMA`. Introduced in [LAAZ11] to cryptanalyse `PRINTcipher` and improved in [LMR15] to break `Robin`, `iSCREAM`, and `Zorro`, these attacks exploit a proper subspace $\mathcal{V}$ of the entire state space that is invariant up to translation under the round functions. This immediately leads to a distinguisher: if initial and final state, i.e. plaintext and ciphertext, belong to certain (possibly different) cosets of $\mathcal{V}$, then the key k (or its subkeys) must belong to a specific coset of $\mathcal{V}$ as well and thus k is a weak key that can be at least brute-forced in time strictly smaller than brute-forcing the whole key space.

Here we only consider the simple type of attack where all the affine subspaces between the rounds are cosets of the same vector subspace $\mathcal{V}$. This is the same setting of the attacks mentioned above. In particular, also `MIDORI` uses different types of rounds.

In Figure 9 we depict the first three rounds of the forward part of `QARMA`, where T_i is the value of the tweak at the i-th round, $0 \leq i \leq r$.

Put $\ell(x) := (\tau \circ M)(x)$ and assume that $(S \circ \ell)(u + \mathcal{V}) = v + \mathcal{V}$ where $\mathcal{V} \subseteq \mathbb{F}_2^{16\,m}$. Since we are assuming that the first short round maps a certain coset $x + \mathcal{V}$ of $\mathcal{V}$ to $v + \mathcal{V}$, we have that ℓ maps $u + \mathcal{V}$ to $x + \mathcal{V}$. Since ℓ is a linear operator not only over $R_m^{4\times 4}$ (where R_m is the ring defined in § 3.1) but also when considered as a mapping of $\mathbb{F}_2^{16\,m}$ onto itself, we have $\ell(u) + \ell(\mathcal{V}) = \ell(u + \mathcal{V}) = x + \mathcal{V}$ for some x, whence $\ell(\mathcal{V}) = \mathcal{V}$ and $x + \ell(u) \in \mathcal{V}$.

We now see that all $c_i + T_i$, $i \geq 1$ in the forward part of `QARMA` must belong to a fixed coset $z + \mathcal{V}$ of $\mathcal{V}$ and therefore all $(c_i + T_i) + (c_1 + T_1) \in \mathcal{V}$, $i \geq 1$ (this also implies $k^0 \in (u + v + z) + \mathcal{V}$). Note that we are not making an assumption on $c_0 + T$ because once the cosets are known to which the core and whitening keys must belong in order to form a weak key, the coset of P is easily derived.

A further observation is that the easiest form of such an attack implies that the transitions in the backward rounds of `QARMA` mirror those of the forward rounds. Hence $\alpha = (c_i + T_i) + (c_i + \alpha + T_i) \in \mathcal{V}$ as well; If the cosets properly chain through the forward rounds, then they automatically chain through the backward rounds as well.

Now, in order to understand the action of τ on the cosets, let us consider the central construction, with coset propagation as in Figure 10. Proceeding from the two sides towards the center, we see that the image of $v + \mathcal{V}$ under τ equals $k^0 + \ell(v) + \mathcal{V}$ – in other words it must be a coset of $\mathcal{V}$, and therefore $\tau(\mathcal{V}) = \mathcal{V}$. Since $\mathcal{V}$ is invariant under τ and $\ell = \tau \circ M$, it is invariant also under M.

This allows us to construct a space $\mathcal{U}$ that must be contained in any invariant subspace

Table 7: Average and Ranges for the Lower Bounds on the Dimensions of Invariant Subspaces of QARMA and MANTIS, Comparing Different Diffusion Matrices

r		4	5	6	7	8
QARMA-64 $/M_{4,2}$	*Average*	54.41	60.32	62.08	63.02	63.51
	Range	[41..58]	[48..62]	[52..63]	[55..64]	[58..64]
QARMA-64 $/M_{4,3}$	*Average*	50.51	57.46	59.95	61.38	62.39
	Range	[36..54]	[45..60]	[51..61]	[53..62]	[56..63]
QARMA-64 $/M_0$	*Average*	38.66	47.15	51.90	54.91	57.18
	Range	[28..41]	[35..51]	[41..54]	[45..56]	[50..58]
MANTIS	*Average*	39.59	46.92	52.93	55.37	57.31
	Range	[32..41]	[38..48]	[46..54]	[50..56]	[51..58]

r		7	8	9	10	11	12
QARMA-128 $/M_{8,2}$	*Average*	122.17	123.61	124.72	125.69	126.51	127.15
	Range	[115..123]	[117..124]	[120..125]	[121..126]	[122..127]	[124..128]
QARMA-128 $/M_{8,3}$	*Average*	105.45	114.01	117.92	120.01	121.50	122.73
	Range	[93..108]	[104..116]	[109..120]	[112..121]	[115..122]	[118..123]
QARMA-128 $/M_0$	*Average*	82.43	92.17	99.61	104.17	107.17	109.58
	Range	[73..84]	[80..94]	[84..104]	[94..106]	[98..108]	[101..110]

$\mathcal{V}$ of the cipher. It is generated by the vectors α, $c_i + c_1 + T_i + T_1$, $1 \leq i < r$, and by the vectors obtained repeatedly applying τ and M to this initial set. If $\mathcal{U}$ is very large in $\mathbb{F}_2^n$, i.e. it has small codimension, then the same is true of any invariant subspace $\mathcal{V}$.

In the case of $\texttt{QARMA}_7$-64, for $T = 0$ this space has dimension 62. For $\texttt{QARMA}_{11}$-128 with $T = 0$ this space has dimension 127. We also computed the dimension of the space $\mathcal{V}$ for one million random values of T for various values of the parameter r, for both $\texttt{QARMA}_r$-64 and $\texttt{QARMA}_r$-128, using the matrices selected in § 3.1 and other candidates, including the MIDORI circulant M_0. The resulting values of the average dimension of $\mathcal{U}$ together with the smallest and largest dimensions found are reported in Table 7.

The smallest values in each range are very rare; calculating $\dim(\mathcal{U})$ for $\texttt{QARMA}_7$-64 for ten million random T's we obtain that the dimensions from 55 to 64 occur respectively 2, 27, 252, 1783, 10338, 60780, 369443, 1748626, 4839951, and 2968798 times. It is likely that smaller values occur, but are even rarer.

Remark 7. This data suggests that the smallest invariant subspace is *at least* almost the whole keyspace for the recommended values of r, and using the selected matrices, essentially giving no weak key classes.

The obvious question now is, what happens if we take into account the action of S as well, for instance using the algorithm presented in [LMR15] taking $\mathcal{U}$ as the *nucleon*? To answer this question, we have combined the code used in [LMR15] with our implementations of the construction of $\mathcal{U}$. So far *we have not been able to find any invariant subspace with codimension larger than 1 regardless of the choice of S-Box and tweak for* $\texttt{QARMA}_r$*-64, resp.* $\texttt{QARMA}_r$*-128 for* $r \geq 4$*, resp.* $r \geq 7$*.*

Remark 8. Using the MIDORI matrix, or the discarded matrices listed in § 3.1.2, we obtain spaces $\mathcal{U}$ of smaller dimension. This is clearly correlated to the presence of rotations, i.e. the more entries are equal to 1, the smaller is $\mathcal{U}$.

The analysis for QARMA-64 with the MIDORI circulant applies with trivial changes to MANTIS as well, and the resulting dimensions are quite close. Interestingly, the spaces found for MANTIS usually have slightly larger dimension that those for QARMA-64 with the

MIDORI matrix, in other words without the ω LFSR the dimensions are larger. The same phenomenon does *not* occur with QARMA-64 (using $M_{4,2}$) and with QARMA-128 (using $M_{8,2}$): The average dimensions obtained removing ω show fluctuations by less than ± 0.2, and are therefore not reported.

Remark 9. Our analysis leaves the possibility that for some very rare tweak values there could be a weak set of keys. It is an open question how to determine these tweaks (if they exist) and how to exploit the weakness of the keys in this case.

Remark 10. To prove bounds for all tweak values seems to be outside the scope of current techniques. An analysis along the lines of the subspace trail cryptanalysis [GRR17, GR16], which would include the cases where the round functions may map cosets of different spaces would shed a better light on the resistance of QARMA against these types of attacks.

4.4 Security Implications of the Central Construction

It is tempting to dismiss the possibility of reflection attacks or attacks on Even-Mansour schemes with involutions simply on the basis that the pseudo-reflector do not constitute an involution. However, the central construction still shows high likelihood self-differentials that depend on k^1, warranting a closer analysis. For simplicity, we use a single matrix M in the analysis of the central construction here to simplify the notation. This does not change the results discussed here.

Besides the whitening and the central key addition, a further crucial difference with respect to MANTIS is the use of two additional ShuffleCells layers. Without them the central group of state transformations formed by M, S, M, addition of k^1, S and M layers would only act independently on four partitions of the state, each consisting of four cells. Indeed, with ShuffleCells, better diffusion is achieved: for instance, in QARMA-64 any single bit of the state on one side of the central construction affects non-linearly at least four bits in each of twelve cells (out of sixteen) on the opposite side, as opposed to just four cells without ShuffleCells – and it affects non-linearly all bits of the state after just one more round, up from all bits in twelve cells. A similar property holds for QARMA-128. These results are proved similarly to Theorem 3.

A tangible effect is that the active S-Box counts in Table 3 for both Class I and II matrices are often larger by a count of 2 or 4 than if we evaluate them without modelling the two ShuffleCells operations.[1]

Related to this is the fact that the central constructions in PRINCE and MANTIS are unkeyed. This means that (in non-differential attacks) guessing the state before the construction gives the state after the construction practically for free. In the case of PRINCE in [RR16] this led to attacks with a complexity slightly lower than the security claim for the full cipher. The middle key addition prevents this type of cryptanalysis as well (and, as explained in Section 5, it does not increase latency).

However, the most important security improvement is regarding (analogues of) reflection attacks. We use Figure 11 as a reference in the following.

The pseudo-reflector *is* an involution if and only if the key k^1 is a fixed point of M (so there are 2^{32} such values for QARMA-64. resp. 2^{72} for QARMA-128), and if $\Delta = 0$ then also $\Delta'' = 0$, but at this point the formation of reflection characteristics is thwarted by the use of the whitening keys in the central rounds, unless $w^0 + w^1 = \alpha$ or 0, which happens for exactly two values of w^0. We conclude that in this setting for only one key K in 2^{95} (for QARMA-64, resp. 2^{183} for QARMA-128) a reflection attack could be attempted:

[1] Also, if we use MILP to count the number of linearly or non-related-tweak differentially active S-Boxes through the two rounds before and the two rounds after the pseudo-reflector we verify that we have at least sixteen active S-Boxes with the ShuffleCells and only eight without. This result holds for both Class I and Class II matrices. This means that the two sub-ciphers of QARMA are better "coupled" by the central rounds when the additional ShuffleCells are included.

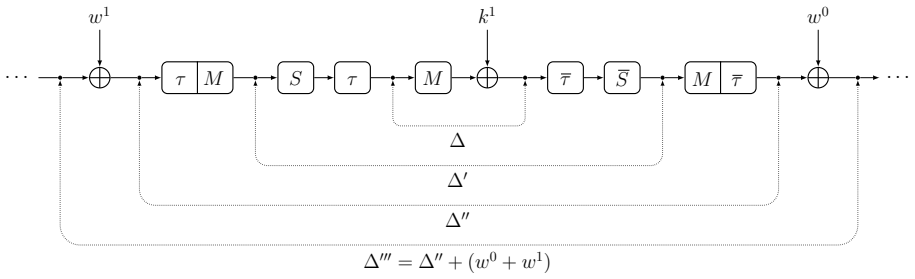

Figure 11: Reflection Self-Differentials

In the case of iterative characteristics, if the palindromic structure begins with a $\Delta''' = \alpha$, the successive differences are 0, then α again, and so on; if the structure begins with a $\Delta''' = 0$, the differences are α, 0, and so on. At the last round the attacker can only try to guess whether the difference has propagated, because of the whitening. Then, the considerations in [SBY+15] would apply, for which, in the case of $\texttt{QARMA}_7$-64, we believe similar attack complexities as in PRINCE to hold (and correspondigly higher complexities would hold for $\texttt{QARMA}_{11}$-128).

If k^1 is not a fixed point of M, i.e. $k^1 \in \ker(M + I)$, then the pseudo-reflector cannot have fixed points, since every such fixed point z would satisfy both $z = M \cdot z + k_1$ and $z = M \cdot z + M \cdot k_1$, which is a contradiction. So attacks based on $\Delta = 0$ cannot be mounted. But this cannot exclude other types of characteristics.

In fact, for any fixed k^1, the self-differential Δ can take only 2^{32} (resp. 2^{56}) equiprobable values, which depend only on the class of k^1 in $R^{4\times4}/\operatorname{Im}(M+I)$. In fact, if $x+Mx+k^1 = \Delta$, then $(x + \ell) + M(x + \ell) + (k^1 + (M + I)\ell) = \Delta$, and there is a 1-1 correspondence $x \leftrightarrow x + \ell$ between the sets of values for which this difference holds for the two keys.

These values Δ can then propagate to values Δ'' with some probability, and for each such Δ'' there are two value of w^0 for which $\Delta''' = \Delta'' + (w^0 + w^1) = \alpha$ or 0. This leads to analogues of attacks based on the second type of middle-round characteristic in [SBY+15]. Now, we see that for each core key, text and tweak, there are two whitening keys in 2^{64} (resp. 2^{128}) that allow a reflection attack to be bootstrapped. So an attacker would attempt to change texts and tweaks hoping that the correct differential Δ'' is hit that leads to $\Delta''' = \alpha$ or 0, and each time verifying whether the differential holds. Unless k^1 is unknown, no assumption on Δ can be done.

The improvement with respect to the attacks on reflection ciphers is that the likelihood of a useful differential Δ''' decreases from 2^{-32} (resp. 2^{-64}) to 2^{-63} (resp. 2^{-127}) and so we get correspondingly reduced success likelihoods – the final probabilities are not as small as our estimates above regarding the $\Delta = 0$ case (in fact, we obtain 2^{-107} and 2^{-189}, respectively), but most likely with data requirements (upwards of 2^{60} for QARMA-64, for instance) that will not compromise the desired level of tradeoff security. Deriving more precise complexity estimations is definitely an important area of research.

Also, the impossibility to control the self-differentials when k^1 and/or w^0 are unknown should prevent analysis based on impossible reflection characteristics.

4.5 Attacks on Even-Mansour Schemes

Recall that the whitening key derivation function $o(\cdot)$ is an orthomorphism. Many of the attacks described in this subsection apply to single-key Even-Mansour schemes. If orthomorphisms are used to create a key schedule, the complexity of the attacks usually increases and approaches that of schemes with independent keys or using independent permutations (see [CLL+14] for the two-round case). We are not claiming that QARMA's

structure offers the security of an EM scheme with independent keys and permutations, but for the attack complexity estimates in the following that are made under a single key assumption, the results are likely not tight. Finally, consider that attacks on EM schemes assume the use of random permutations, whereas the concrete functions we are using are far from that, so the considerations here do not exclude attacks that additionally exploit some property of our permutations, but only apply to generic attacks.

An analogue of the cryptanalysis described in [DDKS13] seems unlikely: The single-key three-round with an involution attack is the one that seems closer to our design. It can be adapted at once observing that, for each fixed core key, the mapping $x \mapsto \Delta$ assumes only $2^{n/2}$ values which occur $2^{n/2}$ times each (with respect to the notation in [DDKS13], this is in-degree t). This leads to a time/data/memory $(T/D/M)$ tradeoff of $TD = 2^{3n/2}$, where the data consists of known texts and memory is online storage. The attack has to be repeated for each candidate core key in order to determine the whitening key as well, so we obtain $T = 2^{7n/4}$, $D = 2^{3n/4}$ and $M = D$. Since evaluations of the sub-ciphers can be done for pairs of core keys $(k^0, k^0 + \alpha)$, the memory usage can be halved. For QARMA-64 this turns into an attack with 2^{112} time and 2^{47} data.

The single-key two-round attack can be applied, under the assumption that for, a known core key, a certain Δ will occur with likelihood $2^{-n/2}$ (but choosing the plaintext does not seem to give control on this event). In this case the cipher collapses into a single-key two-round EM construction, not a single round EM, because $o(\cdot)$ is an orthomorphism. For each core key, time complexity is slightly smaller than 2^n, data (known texts) is slightly smaller than $2^{3n/4}$, and memory is around $2^{n/4}$. The time complexity must be multiplied by 2^n to cover all core keys, and the data by $2^{n/2}$ because of the usable proportion, whereas online memory usage stay the same. We do get an attack with T slightly better than brute force, but $TD \sim 2^{14n/4}$.

These estimates are all better than the corresponding complexities for attacks on the simpler FX-construction used, say, in PRINCE.

Similarly, for the attacks in [DDKS15], with the same likelihoods for a known central difference Δ for a certain core key (resp. class), the equations to solve for the whitening key (and possibly the remaining bits of the core key) would still correspond to the whole cipher minus the central construction, so we do not expect it to be significantly easier than attempting to exploit reflection characteristics.

Finally, we also observe that a three-round, two-key EM scheme, according to [DDKS14] is attacked with a time/data tradeoff of $TD = 2^{2n}$ where $M = D$ (for unkeyed permutations). It is an open question whether our scheme, with a second key derived from the first by means of an orthomorphism offers the same security bound.

4.6 Security Claims

Similarly to MANTIS and PRINCE, for $\texttt{QARMA}_7$-64 and $\texttt{QARMA}_{11}$-128, we claim that they attain n bits of tradeoff security.

The attacks on MANTIS described in [DEKM16] do not apply to QARMA because of the different choices of S-Boxes, the use of the new Almost-MDS matrices and the new central construction. Despite this, we do not claim security against practical attacks already for $\texttt{QARMA}_5$-64 in the sense defined for MANTIS and PRINCE, i.e. that no related-tweak attack should be applicable with less than 2^{30} chosen or 2^{40} known text pairs. We suggest that $\texttt{QARMA}_5$-64 should be used only if the output is heavily truncated (as in the generation of very short tags) or tweak masking/extension is used, and in all other cases the security against practical attacks in the above sense applies first to $\texttt{QARMA}_6$-64.

Regarding QARMA-128, if a 192-bit security level is required, we believe that $\texttt{QARMA}_9$-128 offers it with a substantial security margin. However, we recommend to use $\texttt{QARMA}_r$-128 at the 256 bit security level with $r = 11$, and to use a parameter $r < 11$ only with tweak masking/extension.

4.7 Security for Specific Applications

Software Security. One of the applications of QARMA, as mentioned at the beginning of the paper, is to software security. A common mitigation against vulnerability exploitation techniques such as *return oriented programming* (ROP) and *jump oriented programming* (JOP) consists in the enforcement of *control flow integrity* (CFI) [ABEL05]. CFI ensures that the control flow of the application stays within the control flow graph, as determined ahead of execution. In order to achieve this, several techniques can be employed, including stack canaries and the shadow stack, with potentially high performance costs [DMW15], and even encrypting the stack [TCV04] with expensive cryptographic techniques.

A different approach consists in taking advantage of the fact that some bits are unused in 64-bit pointers, depending on the environment between 3 and 32 bits. We therefore compute a keyed and tweaked tag of the pointer and insert it in its unused bits – where the key is controlled by a higher execution environment and the tweak is a context (e.g. current address/function, or the unique label of a connected component of the call graph). The tagged pointer is then verified before use. For instance, if the pointer is the link register, it is tagged at a function's prologue and it is verified at the function's epilogue before returning to the caller. This gives a low security level per pointer, but guessing more pointer tags becomes increasingly more difficult and significantly raises the bar for the adversary. In particular, only very short ROP chains may be still feasible with a likelihood acceptable for the attacker, and only for the shortest tags – reducing the complexity of the work also for the code auditors.

In our applications these tags are computed by truncating QARMA's output to 32 bits or less. This also means that attacks on QARMA should have complexity lower than 2^{32} in order to speed up software exploitation attacks, and the attacker must be able to read the stack (possibly with invasive hardware techniques) only obtaining a limited number known truncated ciphertexts. In light of our analysis, this seems extremely unlikely.

More details can be found in [ARM16, QPS17].

Memory Encryption. In the introduction we have argued why TBCs are ideally suitable for applications such as memory encryption when the permutation computed by the cipher must be a function not only of the secret key, but also of the memory address and/or a nonce or counter. QARMA was designed to meet the requirements of this use case.

Even if Galois multiplication or another "tweak masking" function is used to mask the tweaking value(s) or to extend the tweak length as described in § 2.9, the resulting additional latency will often be still significantly lower than the expensive key derivation, say, in the XEX mode. Tweak masking can also be used with any instance of QARMA to provide additional hedging against related-tweak and invariant subspace cryptanalysis.

Hashing. Just as the SKEIN hash function [FLS+10] is designed around THREEFISH, we can adapt SKEIN's UBI Chaining Mode to construct a hash function from QARMA. The result is a 64- or 128-bit keyed hash function, which is not acceptable for high-security applications. This is however useful in use cases where the QARMA functionality is already exposed through the ISA and the security goal is to raise the bar for the adversary instead of making an attack absolutely unfeasible.

5 Hardware Implementation

The most important property of implementations of QARMA is the latency, which is strictly correlated to the critical path length, usually measured in NAND gate equivalents (GE).

It is very difficult to provide a fair comparison of different ciphers across multiple papers, since they are often implemented using different synthesis toolchains, libraries,

and manufacturing processes – with 0.18 μm processes at 100 KHz being quite common. The low frequency helps because most symmetric cryptographic algorithms will run in a clock cycle, and pipelining can be avoided, thus levelling the ground, but it might not necessarily reflect the reality of contemporary high-performance hardware.

We are in the fortunate situation where there are few ciphers that (may) target our use cases, so it is not unduly challenging to provide a sufficiently complete comparison. We use a FinFet 7 nm manufacturing process. Our gate library contains different variants of each of the basic gates, differentiated both in number of inputs and drive strength. The GE number is computed with respect to the *smallest* NAND gate in the library, the standard 4T cell. The area ratios between the various standard gates may differ from other, more commonly used libraries: For instance, whereas XOR = XNOR = 2 NAND as usual for the timing path, we have XOR = XNOR = 2.75 NAND for the area. Besides 3- and 4-way classical gates, the library contains several composite gates, such as $(x, y, z) \mapsto (x \wedge y) \vee z$ and $(x, y, z) \mapsto x \wedge (y \vee z)$, also with negated inputs.

We first report on our implementations of QARMA-64, PRINCE and MANTIS. The results are given in Table 8. All our implementations allow registers and pipeline stage boundaries to be placed between any two rounds, including the central construction, using parameters set at synthesis time. For instance, $\texttt{QARMA}_8$-64 can be instantiated as a single cycle implementation for low frequencies, but it can also be split into two, three, or any number of stages up to 18, in order to be pipelined at very high frequencies.

We synthesised each algorithm twice, targeting minimum area and minimum delay, for unrolled, non-pipelined single cycle operations. In order to achieve this and to ease the synthesis process, we kept clock frequencies low with respect to production runs (which require pipelining). Minimum area was always attained at 100 Mhz and synthesis was relatively fast – usually less than 1000 core seconds on a dedicated server. However, in the minimum delay case the synthesis software had to perform several optimisations, and synthesising one instance of QARMA-64 easily took more than 8000 core seconds.

In Table 9 we report results for QARMA-128 and the AES. Because of the larger synthesis times, we have considered individual rounds and then stitched the circuits together for QARMA-128. As a result the results could likely be slightly, but not significantly, improved. The AES-128 implementation was already available, however we note that it is pipelined, and that is why we decided to report also the latency of a single full round. The full AES delays we list in Table 9 are extrapolations based on the latency of each component in the implementation, and they are only an approximation of the delays the ciphers would have if pipelining were to be removed.

One notices immediately that for the minimum area synthesis results the area in GE of PRINCE and MANTIS is bigger than the values given for instance, in [BCG+12] and [BJK+16], and the difference is considerably bigger for MANTIS. There are a few possible reasons for this discrepancy:

1. The geometries are much smaller than in the processes usually reported in the literature on lightweight ciphers. As a result the synthesis software needs to place more gates with high driving strength, which have larger area, to counter leakage.

2. The ratio of the area of XOR/XNOR gates to NAND gates is higher than in other libraries. As a result, the diffusion layer and the (twea)key addition carry more relative weight. This is only partially offset by the use of smaller combined gates.

3. At finer processes, wiring also has a larger relative area impact, and, besides the data obfuscation path and the key schedule, tweakable ciphers need to carry along the tweak as well.

The last two points also explain the fact that the area difference between PRINCE and MANTIS is in our implementations larger than in [BJK+16].

Table 8: Delay and Area for QARMA-64, MANTIS and PRINCE, all fully unrolled

Targeting	*Minimum Area*			*Minimum Delay*		
	Delay	Area		Delay	Area	
Cipher	*ns*	μm^2	GE	*ns*	μm^2	GE
$\texttt{QARMA}_5$-64-σ_0	4.52	733.0	13395	2.20	1238.1	22626
$\texttt{QARMA}_6$-64-σ_0	5.28	836.7	14984	2.45	1403.0	25640
$\texttt{QARMA}_7$-64-σ_0	6.04	953.0	17109	2.75	1595.3	29154
$\texttt{QARMA}_8$-64-σ_0	6.59	1053.3	18942	3.14	1778.9	32637
$\texttt{QARMA}_5$-64-σ_1	5.07	752.5	13751	2.43	1406.3	25700
$\texttt{QARMA}_6$-64-σ_1	5.80	882.4	16126	2.84	1592.7	29106
$\texttt{QARMA}_7$-64-σ_1	6.23	1004.8	18362	3.25	1879.9	34354
$\texttt{QARMA}_8$-64-σ_1	6.63	1125.9	20575	3.56	2065.6	37749
$\texttt{QARMA}_5$-64-σ_2	4.96	791.9	14472	2.42	1474.0	26937
$\texttt{QARMA}_6$-64-σ_2	5.69	919.0	16795	2.84	1704.1	31142
$\texttt{QARMA}_7$-64-σ_2	6.17	1042.7	19055	3.19	1931.4	35296
$\texttt{QARMA}_8$-64-σ_2	6.61	1153.2	21075	3.60	2145.9	39216
$\texttt{MANTIS}_5$	4.41	660.7	12075	2.20	1152.3	21058
$\texttt{MANTIS}_6$	5.11	764.4	13968	2.56	1359.1	24837
$\texttt{MANTIS}_7$	5.85	866.3	15831	2.94	1532.1	27998
$\texttt{MANTIS}_8$	6.48	956.2	17474	3.33	1699.3	31054
PRINCE	4.07	476.2	8703	2.12	1119.8	20464
Mult. in $\mathbb{F}_{2^{64}}$	1.05	715.9	13083	0.44	924.6	16897

A surprising outcome of the implementations is that the versions of QARMA based on σ_2 often slightly outperform those based on σ_1, whereas the latter consistently have slightly smaller area. This means that the synthesis tools performed some optimisations on the σ_2 variants that were not obvious at the time of the S-Box design.

Note also that the delays of MANTIS and QARMA-64 using the S-Box σ_0 are quite close, despite the fact that the middle key addition is on the critical path of the data obfuscation path. In fact, this key addition does not add latency, since it follows a matrix multiplication, that for each output bit needs to compute two XORs, which are cascaded, resulting in a circuit with a depth of two XORs. The key addition is a third XOR, that can be computed in parallel with the first one, so we obtain a tree of three XORs with the same depth of two XORs. In any case, QARMA-64 is latency-wise competitive with respect to MANTIS also when other S-Boxes are used, while (possibly) offering better hedging against various types of cryptanalysis.

With respect to PRINCE, the most obvious comparison is done pitting a single $\texttt{QARMA}_7$-64 instance against PRINCE used in a XEX-like construction where PRINCE's whitening key is derived using a second PRINCE instance and Galois multiplication is used on the whitening key for each following block in the same cache line. Even using QARMA's largest implementation (using the σ_2 S-Box) with QARMA we have a latency of 6.17 ns for the first block for a total area of 19055 GE, whereas with PRINCE the latency is at least 8.14 ns and the circuit has a total area of $8703 \times 2 + 13083 = 30489$ GE (including the Galois multiplication). If both the whitening and core PRINCE keys depend on the tweak, as in [BDA15], we may need a total of *three* PRINCE instances and one Galois multiplier, bring the area requirement to $8703 \times 3 + 13083 = 39192$ GE.

Hence, in a complete solution QARMA-64 has both an area and a latency advantage with respect to a solutions designed around a non-tweakable cipher. With the 128-bit versions (comparing with the AES) the area and latency advantages increase further.

Table 9: Delay and Area for QARMA-128 (fully unrolled), and the AES (pipelined)

Targeting	*Minimum Area*			*Minimum Delay*		
	Delay	Area		Delay	Area	
Cipher	*ns*	μm^2	GE	*ns*	μm^2	GE
$\texttt{QARMA}_8$-128-σ_0	6.33	2096.2	38309	3.42	3463.4	63293
$\texttt{QARMA}_9$-128-σ_0	7.15	2321.4	42423	3.79	3844.6	70260
$\texttt{QARMA}_{10}$-128-σ_0	7.95	2528.2	46203	4.16	4225.8	77226
$\texttt{QARMA}_{11}$-128-σ_0	8.71	2732.2	49931	4.53	4607.0	84192
$\texttt{QARMA}_8$-128-σ_1	6.70	2221.1	40591	3.62	3995.2	73012
$\texttt{QARMA}_9$-128-σ_1	7.43	2461.0	44975	4.03	4430.6	80969
$\texttt{QARMA}_{10}$-128-σ_1	8.15	2707.2	49475	4.40	4866.0	88926
$\texttt{QARMA}_{11}$-128-σ_1	8.88	2947.9	53872	4.80	5301.4	96883
$\texttt{QARMA}_8$-128-σ_2	6.65	2301.6	42062	3.76	4178.4	76360
$\texttt{QARMA}_9$-128-σ_2	7.38	2552.4	46645	4.18	4632.5	84658
$\texttt{QARMA}_{10}$-128-σ_2	8.12	2804.2	51246	4.60	5087.7	92977
$\texttt{QARMA}_{11}$-128-σ_2	8.84	3027.4	55325	4.99	5520.3	100883
AES-128, pipelined	15.67	3894.1	71164	*Note: The latency of one*		
AES-256, pipelined	21.99	5533.7	101128	*full AES round is* 1.58 *ns*		

6 Conclusions and Open Questions

We have introduced QARMA, a new lightweight TBC family that comes in 64 and 128 bit block and tweak sizes, aimed at 128 and 256 bit levels of security, respectively.

QARMA-64 is the standard block cipher used in the ARMv8.3-A ISA extensions for Control Flow Integrity [ARM16], which have been designed by Qualcomm's Product Security Team [AKT14, QPS17].

A memory encryption engine designed around QARMA can offer both area and latency advantages with respect to solutions designed around non-tweakable ciphers.

The cipher's security has been analysed and we believe it offers reasonable security margins with the recommended numbers of rounds. However, there is need for more cryptanalysis of the integral and structural types, including subspace trail cryptanalysis.

We believe that QARMA can spur research into the analysis of reflection-like differentials in ciphers that are symmetric around a non-involutory structure. Another interesting open question is how to design MILP models for counting active S-Boxes in order to attain better bounds on the number of the active 4-bit halves in QARMA-128's (and MIDORI-128's) 8-bit S-Boxes – without resorting to models which are bit-wise throughout which quickly become impractical.

Finally, the tweak masking/extension technique proposed in § 2.9 still needs to be properly analysed.

Acknowledgements

We thank the anonymous reviewers for their insightful comments and suggestions. We express gratitude to the authors of [BJK+16] for the fruitful interaction, in particular to Christof Beierle for sharing his MILP modelling of the Class I state transitions used in § 3.1.2. Scott McGregor deserves a special mention for teaching us hardware design, fixing our Verilog, and running the synthesis. This work would not have been possible without the full support of our management chain: Alex Gantman, Renwei Ge, and Brian Rosenberg. We acknowledge many interesting discussions with Can Acar, Satish Anand, Christina Boura, Antonio Cardoso Costa, Xiaoyang Dong, Orr Dunkelman,

Richard Grisenthwaite, Rene Peralta, Meltem Sönmez Turan, and Robert Turner. We are grateful to Alex Dent for proofreading, and indebted to the Twitter account @FakeIACR for good laughs during boring programming sessions.

References

[ABEL05] Martín Abadi, Mihai Budiu, Úlfar Erlingsson, and Jay Ligatti. Control-flow integrity. In Vijay Atluri, Catherine A. Meadows, and Ari Juels, editors, *Proceedings of the 12th ACM Conference on Computer and Communications Security, CCS 2005, Alexandria, VA, USA, November 7-11, 2005*, pages 340–353. ACM, 2005.

[ADK+14] Martin R. Albrecht, Benedikt Driessen, Elif Bilge Kavun, Gregor Leander, Christof Paar, and Tolga Yalçin. Block Ciphers - Focus on the Linear Layer (feat. PRIDE). In Garay and Gennaro [GG14], pages 57–76.

[AKT14] Can Acar, Arvind Krishnaswamy, and Robert Turner. Code pointer authentication for hardware flow control, October 2014. United States Patent US9514305 B2. Assignee: QUALCOMM Incorporated.

[ARM16] ARM Connected blog. ARMv8-A architecture – 2016 additions. https://www.community.arm.com/processors/b/blog/posts/armv8-a-architecture-2016-additions, October 2016.

[BBI+15] Subhadeep Banik, Andrey Bogdanov, Takanori Isobe, Kyoji Shibutani, Harunaga Hiwatari, Toru Akishita, and Francesco Regazzoni. Midori: A Block Cipher for Low Energy. In Tetsu Iwata and Jung Hee Cheon, editors, *Advances in Cryptology - ASIACRYPT 2015 - 21st International Conference on the Theory and Application of Cryptology and Information Security, Auckland, New Zealand, November 29 - December 3, 2015, Proceedings, Part II*, volume 9453 of *Lecture Notes in Computer Science*, pages 411–436. Springer, 2015.

[BCC11] Christina Boura, Anne Canteaut, and Christophe De Cannière. Higher-Order Differential Properties of Keccak and *Luffa*. In Antoine Joux, editor, *Fast Software Encryption - 18th International Workshop, FSE 2011, Lyngby, Denmark, February 13-16, 2011, Revised Selected Papers*, volume 6733 of *Lecture Notes in Computer Science*, pages 252–269. Springer, 2011.

[BCG+12] Julia Borghoff, Anne Canteaut, Tim Güneysu, Elif Bilge Kavun, Miroslav Knezevic, Lars R. Knudsen, Gregor Leander, Ventzislav Nikov, Christof Paar, Christian Rechberger, Peter Rombouts, Søren S. Thomsen, and Tolga Yalçin. PRINCE - A Low-latency Block Cipher for Pervasive Computing Applications (Full version). *IACR Cryptology ePrint Archive*, 2012:529, 2012.

[BDA15] Billy Bob Brumley, Vinoth Kumar Deivasigamani, and Satish Nithianandan Anand. Dynamic encryption keys for use with XTS encryption systems employing reduced-round ciphers, September 2015. United States Patent Application 20150261965 A1. Assignee: QUALCOMM Incorporated.

[BJK+16] Christof Beierle, Jérémy Jean, Stefan Kölbl, Gregor Leander, Amir Moradi, Thomas Peyrin, Yu Sasaki, Pascal Sasdrich, and Siang Meng Sim. The SKINNY Family of Block Ciphers and Its Low-Latency Variant MANTIS. In Matthew Robshaw and Jonathan Katz, editors, *Advances in Cryptology - CRYPTO 2016 - 36th Annual International Cryptology Conference, Santa*

Barbara, CA, USA, August 14-18, 2016, Proceedings, Part II, volume 9815 of *Lecture Notes in Computer Science*, pages 123–153. Springer, 2016.

[BKL+07] Andrey Bogdanov, Lars R. Knudsen, Gregor Leander, Christof Paar, Axel Poschmann, Matthew J. B. Robshaw, Yannick Seurin, and C. Vikkelsoe. PRESENT: An Ultra-Lightweight Block Cipher. In Pascal Paillier and Ingrid Verbauwhede, editors, *CHES 2007*, volume 4727 of *Lecture Notes in Computer Science*, pages 450–466. Springer, 2007.

[BSS+13] Ray Beaulieu, Douglas Shors, Jason Smith, Stefan Treatman-Clark, Bryan Weeks, and Louis Wingers. The SIMON and SPECK Families of Lightweight Block Ciphers. Cryptology ePrint Archive, Report 2013/404, http://eprint.iacr.org/2013/404, 2013.

[CDK09] Christophe De Cannière, Orr Dunkelman, and Miroslav Knezevic. KATAN and KTANTAN - A Family of Small and Efficient Hardware-Oriented Block Ciphers. In Christophe Clavier and Kris Gaj, editors, *CHES 2009*, volume 5747 of *Lecture Notes in Computer Science*, pages 272–288. Springer, 2009.

[CK08] Jiali Choy and Khoongming Khoo. New applications of differential bounds of the SDS structure. In Tzong-Chen Wu, Chin-Laung Lei, Vincent Rijmen, and Der-Tsai Lee, editors, *Information Security, 11th International Conference, ISC 2008, Taipei, Taiwan, September 15-18, 2008. Proceedings*, volume 5222 of *Lecture Notes in Computer Science*, pages 367–384. Springer, 2008. Corrected version at http://eprint.iacr.org/2008/395.

[CLL+14] Shan Chen, Rodolphe Lampe, Jooyoung Lee, Yannick Seurin, and John P. Steinberger. Minimizing the two-round Even-Mansour cipher. In Garay and Gennaro [GG14], pages 39–56.

[CP02] Nicolas Courtois and Josef Pieprzyk. Cryptanalysis of block ciphers with overdefined systems of equations. In Yuliang Zheng, editor, *ASIACRYPT 2002*, volume 2501 of *Lecture Notes in Computer Science*, pages 267–287. Springer, 2002.

[Cro00] Paul Crowley. Mercy: A fast large block cipher for disk sector encryption. In Bruce Schneier, editor, *Fast Software Encryption, 7th International Workshop, FSE 2000, New York, NY, USA, April 10-12, 2000, Proceedings*, volume 1978 of *Lecture Notes in Computer Science*, pages 49–63. Springer, 2000.

[DDKS13] Itai Dinur, Orr Dunkelman, Nathan Keller, and Adi Shamir. Key Recovery Attacks on 3-round Even-Mansour, 8-step LED-128, and Full AES^2. In Sako and Sarkar [SS13], pages 337–356.

[DDKS14] Itai Dinur, Orr Dunkelman, Nathan Keller, and Adi Shamir. Cryptanalysis of iterated Even-Mansour schemes with two keys. In Palash Sarkar and Tetsu Iwata, editors, *Advances in Cryptology - ASIACRYPT 2014 - 20th International Conference on the Theory and Application of Cryptology and Information Security, Kaoshiung, Taiwan, R.O.C., December 7-11, 2014. Proceedings, Part I*, volume 8873 of *Lecture Notes in Computer Science*, pages 439–457. Springer, 2014.

[DDKS15] Itai Dinur, Orr Dunkelman, Nathan Keller, and Adi Shamir. Reflections on slide with a twist attacks. *Des. Codes Cryptography*, 77(2-3):633–651, 2015.

[DEKM16] Christoph Dobraunig, Maria Eichlseder, Daniel Kales, and Florian Mendel. Practical key recovery attack on MANTIS-5. *IACR Cryptology ePrint Archive*, 2016:754, 2016.

[Din15] Itai Dinur. Cryptanalytic Time-Memory-Data tradeoffs for FX-constructions with applications to PRINCE and PRIDE. In Oswald and Fischlin [OF15], pages 231–253.

[DMW15] Thurston H. Y. Dang, Petros Maniatis, and David Wagner. The performance cost of shadow stacks and stack canaries. In Feng Bao, Steven Miller, Jianying Zhou, and Gail-Joon Ahn, editors, *Proceedings of the 10th ACM Symposium on Information, Computer and Communications Security, ASIA CCS '15, Singapore, April 14-17, 2015*, pages 555–566. ACM, 2015.

[DR02] Joan Daemen and Vincent Rijmen. *The Design of Rijndael: AES - The Advanced Encryption Standard*. Information Security and Cryptography. Springer, 2002. http://dx.doi.org/10.1007/978-3-662-04722-4.

[DRS+14] S. M. Dehnavi, A. Mahmoodi Rishakani, M. R. Mirzaee Shamsabad, Hamidreza Maimani, and Einollah Pasha. Construction of new families of MDS diffusion layers. Cryptology ePrint Archive, Report 2014/011, 2014. http://eprint.iacr.org/.

[DRS15] S. M. Dehnavi, A. Mahmoodi Rishakani, and M. R. Mirzaee Shamsabad. Bitwise Linear Mappings with Good Cryptographic Properties and Efficient Implementation. Cryptology ePrint Archive, Report 2015/225, 2015. http://eprint.iacr.org/.

[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, *ASIACRYPT '91*, volume 739 of *Lecture Notes in Computer Science*, pages 210–224. Springer, 1991.

[EM97] Shimon Even and Yishay Mansour. A Construction of a Cipher from a Single Pseudorandom Permutation. *Journal of Cryptology*, 10(3):151–162, 1997.

[FLS+10] Niels Ferguson, Stefan Lucks, Bruce Schneier, Doug Whiting, Mihir Bellare, Tadayoshi Kohno, Jon Callas, and Jesse Walker. The Skein Hash Function Family. http://www.skein-hash.info, October 2010.

[GG14] Juan A. Garay and Rosario Gennaro, editors. *Advances in Cryptology - CRYPTO 2014 - 34th Annual Cryptology Conference, Santa Barbara, CA, USA, August 17-21, 2014, Proceedings, Part I*, volume 8616 of *Lecture Notes in Computer Science*. Springer, 2014.

[GJN+17] Jian Guo, Jérémy Jean, Ivica Nikolic, Kexin Qiao, Yu Sasaki, and Siang Meng Sim. Invariant Subspace Attack Against Midori64 and The Resistance Criteria for S-box Designs. *Transaction on Symmetric Cryptanalysis (FSE 2017)*, 2017.

[GNL11] Zheng Gong, Svetla Nikova, and Yurl Wei Law. KLEIN: A New Family of Lightweight Block Ciphers. In Ari Juels and Christof Paar, editors, *RFIDSec 2011*, volume 7055 of *Lecture Notes in Computer Science*, pages 1–18. Springer, 2011.

[GPPR12] Jian Guo, Thomas Peyrin, Axel Poschmann, and Matthew J. B. Robshaw. The LED Block Cipher. *IACR Cryptology ePrint Archive*, 2012:600, 2012.

[GR16] Lorenzo Grassi and Christian Rechberger. Practical low data-complexity subspace-trail cryptanalysis of round-reduced PRINCE. In Orr Dunkelman and Somitra Kumar Sanadhya, editors, *Progress in Cryptology - INDOCRYPT 2016 - 17th International Conference on Cryptology in India, Kolkata, India, December 11-14, 2016, Proceedings*, volume 10095 of *Lecture Notes in Computer Science*, pages 322–342, 2016.

[Gro96] Lov K. Grover. A fast quantum mechanical algorithm for database search. In Gary L. Miller, editor, *Proceedings of the Twenty-Eighth Annual ACM Symposium on the Theory of Computing, Philadelphia, Pennsylvania, USA, May 22-24, 1996*, pages 212–219. ACM, 1996.

[GRR17] Lorenzo Grassi, Christian Rechberger, and Sondre Rønjom. Subspace Trail Cryptanalysis and its Applications to AES. *Transaction on Symmetric Cryptanalysis (FSE 2017)*, 2017.

[Gue16] Shay Gueron. Intel's SGX Memory Encryption Engine (Slides). https://drive.google.com/file/d/0Bzm_4XrWnl5zOXdTcUlEMmdZem8/view, 2016.

[HT13] Michael Henson and Stephen Taylor. Memory encryption: A survey of existing techniques. *ACM Comput. Surv.*, 46(4):53:1–53:26, 2013.

[JNP14] Jérémy Jean, Ivica Nikolic, and Thomas Peyrin. Tweaks and Keys for Block Ciphers: The TWEAKEY Framework. In Palash Sarkar and Tetsu Iwata, editors, *Advances in Cryptology - ASIACRYPT 2014 - 20th International Conference on the Theory and Application of Cryptology and Information Security, Kaoshiung, Taiwan, R.O.C., December 7-11, 2014, Proceedings, Part II*, volume 8874 of *Lecture Notes in Computer Science*, pages 274–288. Springer, 2014.

[Kar07] Orhun Kara. Reflection attacks on product ciphers. Cryptology ePrint Archive, Report 2007/043, 2007. http://eprint.iacr.org/2007/043.

[KD79] John B. Kam and George I. Davida. Structured design of substitution-permutation encryption networks. *IEEE Transactions on Computers*, 28(10):747–753, October 1979.

[LAAZ11] Gregor Leander, Mohamed Ahmed Abdelraheem, Hoda AlKhzaimi, and Erik Zenner. A Cryptanalysis of PRINTcipher: The Invariant Subspace Attack. In Phillip Rogaway, editor, *Advances in Cryptology - CRYPTO 2011 - 31st Annual Cryptology Conference, Santa Barbara, CA, USA, August 14-18, 2011. Proceedings*, volume 6841 of *Lecture Notes in Computer Science*, pages 206–221. Springer, 2011.

[LMR15] Gregor Leander, Brice Minaud, and Sondre Rønjom. A Generic Approach to Invariant Subspace Attacks: Cryptanalysis of Robin, iSCREAM and Zorro. In Oswald and Fischlin [OF15], pages 254–283.

[LN14] Virginie Lallemand and María Naya-Plasencia. Cryptanalysis of KLEIN. In Carlos Cid and Christian Rechberger, editors, *Fast Software Encryption - 21st International Workshop, FSE 2014, London, UK, March 3-5, 2014. Revised Selected Papers*, volume 8540 of *Lecture Notes in Computer Science*, pages 451–470. Springer, 2014.

[LP07] Gregor Leander and Axel Poschmann. On the Classification of 4-Bit S-Boxes. In Claude Carlet and Berk Sunar, editors, *Arithmetic of Finite Fields, First*

International Workshop, WAIFI 2007, Madrid, Spain, June 21-22, 2007, Proceedings, volume 4547 of *Lecture Notes in Computer Science*, pages 159–176. Springer, 2007.

[LRW02] Moses Liskov, Ronald L. Rivest, and David Wagner. Tweakable Block Ciphers. In Moti Yung, editor, *CRYPTO 2002*, volume 2442 of *Lecture Notes in Computer Science*, pages 31–46. Springer, 2002.

[LST12] Will Landecker, Thomas Shrimpton, and R. Seth Terashima. Tweakable Blockciphers with Beyond Birthday-Bound Security. In Reihaneh Safavi-Naini and Ran Canetti, editors, *Advances in Cryptology - CRYPTO 2012 - 32nd Annual Cryptology Conference, Santa Barbara, CA, USA, August 19-23, 2012. Proceedings*, volume 7417 of *Lecture Notes in Computer Science*, pages 14–30. Springer, 2012.

[Mar10] Luther Martin. XTS: A mode of AES for encrypting hard disks. *IEEE Security & Privacy*, 8(3):68–69, 2010.

[McC56] Edward J. McCluskey. Minimization of Boolean Functions. *Bell System Technical Journal*, 35(6):1417–1444, November 1956.

[MSGR10] Marcel Medwed, François-Xavier Standaert, Johann Großschädl, and Francesco Regazzoni. Fresh re-keying: Security against side-channel and fault attacks for low-cost devices. In Daniel J. Bernstein and Tanja Lange, editors, *Progress in Cryptology - AFRICACRYPT 2010, Third International Conference on Cryptology in Africa, Stellenbosch, South Africa, May 3-6, 2010. Proceedings*, volume 6055 of *Lecture Notes in Computer Science*, pages 279–296. Springer, 2010.

[MWGP11] Nicky Mouha, Qingju Wang, Dawu Gu, and Bart Preneel. Differential and linear cryptanalysis using mixed-integer linear programming. In Chuankun Wu, Moti Yung, and Dongdai Lin, editors, *Information Security and Cryptology - 7th International Conference, Inscrypt 2011, Beijing, China, November 30 - December 3, 2011. Revised Selected Papers*, volume 7537 of *Lecture Notes in Computer Science*, pages 57–76. Springer, 2011.

[NIS16] NIST. Implementation Guidance for FIPS PUB 140-2 and the Cryptographic Module Validation Program. http://csrc.nist.gov/groups/STM/cmvp/documents/fips140-2/FIPS1402IG.pdf, January 2016.

[OF15] Elisabeth Oswald and Marc Fischlin, editors. *Advances in Cryptology - EUROCRYPT 2015 - 34th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Sofia, Bulgaria, April 26-30, 2015, Proceedings, Part I*, volume 9056 of *Lecture Notes in Computer Science*. Springer, 2015.

[Pri10] Cyril Prissette. An Algorithm to List All the Fixed-Point Free Involutions on a Finite Set. http://arxiv.org/pdf/1006.3993v1.pdf, 2010.

[QPS17] Qualcomm Product Security. Pointer Authentication on ARMv8.3 – Design and Analysis of the New Software Security Instructions. https://www.qualcomm.com/documents/whitepaper-pointer-authentication-armv83, January 2017.

[Qui52] Willard Van Orman Quine. The Problem of Simplifying Truth Functions. *The American Mathematical Monthly*, pages 521–531, October 1952.

[RDS+14] A. Mahmoodi Rishakani, S. M. Dehnavi, M. R. Mirzaee Shamsabad, Hamidreza Maimani, and Einollah Pasha. New concepts in design of lightweight MDS diffusion layers. *11th International ISC Conference on Information Security and Cryptology, 2014*, 2014. DOI: 10.1109/ISCISC.2014.6994017.

[Rog96] Phillip Rogaway. The Security of DESX. *Cryptobytes*, pages 8–11, Summer 1996.

[Rog04] Phillip Rogaway. Efficient Instantiations of Tweakable Blockciphers and Refinements to Modes OCB and PMAC. In Pil Joong Lee, editor, *Advances in Cryptology - ASIACRYPT 2004, 10th International Conference on the Theory and Application of Cryptology and Information Security, Jeju Island, Korea, December 5-9, 2004, Proceedings*, volume 3329 of *Lecture Notes in Computer Science*, pages 16–31. Springer, 2004.

[RR16] Shahram Rasoolzadeh and Håvard Raddum. Cryptanalysis of PRINCE with minimal data. Cryptology ePrint Archive, Report 2016/080, 2016. http://eprint.iacr.org/2016/080.

[Saa11] Markku-Juhani O. Saarinen. Cryptographic Analysis of All 4 × 4-Bit S-Boxes. In Ali Miri and Serge Vaudenay, editors, *Selected Areas in Cryptography - 18th International Workshop, SAC 2011, Toronto, ON, Canada, August 11-12, 2011, Revised Selected Papers*, volume 7118 of *Lecture Notes in Computer Science*, pages 118–133. Springer, 2011.

[SBY+15] Hadi Soleimany, Céline Blondeau, Xiaoli Yu, Wenling Wu, Kaisa Nyberg, Huiling Zhang, Lei Zhang, and Yanfeng Wang. Reflection cryptanalysis of PRINCE-like ciphers. *Journal of Cryptology*, 28(3):718–744, 2015.

[SCG+03] G. Edward Suh, Dwaine E. Clarke, Blaise Gassend, Marten van Dijk, and Srinivas Devadas. Efficient Memory Integrity Verification and Encryption for Secure Processors. In *Proceedings of the 36th Annual International Symposium on Microarchitecture, San Diego, CA, USA, December 3-5, 2003*, pages 339–350. ACM/IEEE Computer Society, 2003.

[Sch99] Rich Schroeppel. Hasty Pudding Cipher specification. http://richard.schroeppel.name:8015/hpc/hpc-spec, May 1999.

[SLG+16] Bing Sun, Meicheng Liu, Jian Guo, Vincent Rijmen, and Ruilin Li. Provable security evaluation of structures against impossible differential and zero correlation linear cryptanalysis. In Marc Fischlin and Jean-Sébastien Coron, editors, *Advances in Cryptology - EUROCRYPT 2016 - 35th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Vienna, Austria, May 8-12, 2016, Proceedings, Part I*, volume 9665 of *Lecture Notes in Computer Science*, pages 196–213. Springer, 2016.

[SS13] Kazue Sako and Palash Sarkar, editors. *Advances in Cryptology - ASIACRYPT 2013 - 19th International Conference on the Theory and Application of Cryptology and Information Security, Bengaluru, India, December 1-5, 2013, Proceedings, Part I*, volume 8269 of *Lecture Notes in Computer Science*. Springer, 2013.

[SS17] Sumanta Sarkar and Habeeb Syed. Lightweight Diffusion Layer: Importance of Toeplitz Matrices. *Transaction on Symmetric Cryptanalysis (FSE 2017)*, 2017.

[SSA+07] Taizo Shirai, Kyoji Shibutani, Toru Akishita, Shiho Moriai, and Tetsu Iwata. The 128-Bit blockcipher CLEFIA. In Alex Biryukov, editor, *FSE 2007*, volume 4593 of *Lecture Notes in Computer Science*, pages 181–195. Springer, 2007.

[ST13] Thomas Shrimpton and R. Seth Terashima. A Modular Framework for Building Variable-Input-Length Tweakable Ciphers. In Sako and Sarkar [SS13], pages 405–423.

[TCV04] Nathan Tuck, Brad Calder, and George Varghese. Hardware and binary modification support for code pointer protection from buffer overflow. In *37th Annual International Symposium on Microarchitecture (MICRO-37 2004), 4-8 December 2004, Portland, OR, USA*, pages 209–220. IEEE Computer Society, 2004.

[WT85] A.F. Webster and Stafford E. Tavares. On the design of s-boxes. In Hugh C. Williams, editor, *CRYPTO '85*, volume 218 of *Lecture Notes in Computer Science*, pages 523–534. Springer, 1985.

[YEP+06] Chenyu Yan, Daniel Englender, Milos Prvulovic, Brian Rogers, and Yan Solihin. Improving Cost, Performance, and Security of Memory Encryption and Authentication. In *33rd International Symposium on Computer Architecture (ISCA 2006), June 17-21, 2006, Boston, MA, USA*, pages 179–190. IEEE Computer Society, 2006.

[ZD16] Rui Zong and Xiaoyang Dong. Meet-in-the-middle attack on QARMA block cipher. Cryptology ePrint Archive, Report 2016/1160, 2016. http://eprint.iacr.org/2016/1160.

A Test Vectors

We include here test vectors for all versions of QARMA, including legacy ones.

A.1 QARMA-64 with $M = Q = M_{4,2}$

```
P  = fb623599da6e8127    T  = 477d469dec0b8762
w0 = 84be85ce9804e94b    k0 = ec2802d4e0a488e9
```

Using the S-Box σ_0.

```
QARMA64_5
C = 3ee99a6c82af0c38

QARMA64_6
C = 9f5c41ec525603c9

QARMA64_7
C = bcaf6c89de930765
```

Using the S-Box σ_1.

```
QARMA64_5
C = 544b0ab95bda7c3a

QARMA64_6
C = a512dd1e4e3ec582

QARMA64_7
C = edf67ff370a483f2
```

Using the S-Box σ_2.

```
QARMA64_5
C = c003b93999b33765

QARMA64_6
C = 270a787275c48d10

QARMA64_7
C = 5c06a7501b63b2fd
```

A.2 QARMA-128 with $M = Q = M_{8,2}$

```
\begin{verbatim}
P  = 2fdbb6a2c395e959 fdfa964e98c1a2e7    T  = 242767fd3486a96d fe9c904be82756a2
w0 = 58948b7ef8e5ec7e f9f8f014de0924eb    k0 = 4e7ce2081002eda4 fe20aaa4d868be09
```

Using the S-Box σ_0.

```
QARMA128_9
P = 2fdbb6a2c395e959 fdfa964e98c1a2e7
C = 81c82ecc34cd73df d221ec5485dc6914

QARMA128_10
P = 2fdbb6a2c395e959 fdfa964e98c1a2e7
C = b9f9d94168846c8c ce53883f43cb747b

QARMA128_11
P = 2fdbb6a2c395e959 fdfa964e98c1a2e7
C = 4fb6e28d37fa1bee c3c75c071674cd3c
```

Using the S-Box σ_1.

```
QARMA128_9
P = 2fdbb6a2c395e959 fdfa964e98c1a2e7
C = e73be2739636da81 d6b5b9dbdd2a9254

QARMA128_10
P = 2fdbb6a2c395e959 fdfa964e98c1a2e7
C = 2c4a4f3d45fd9e09 74b12238e0f469ce

QARMA128_11
P = 2fdbb6a2c395e959 fdfa964e98c1a2e7
C = 2d548c30ba8bf609 5434fb10681bc829
```

Using the S-Box σ_2.

```
QARMA128_9
P = 2fdbb6a2c395e959 fdfa964e98c1a2e7
C = 33e2698893f1903a 93169b32d1ba3cb1

QARMA128_10
P = 2fdbb6a2c395e959 fdfa964e98c1a2e7
C = 4674845c356a8f85 066f74e93ed65f55

QARMA128_11
P = 2fdbb6a2c395e959 fdfa964e98c1a2e7
C = 272827c10d18fb6e e76d4284ecdb41d1
```

A.3 QARMA-64 with $M = Q = M_{4,1}$ (legacy)

```
P  = fb623599da6e8127    T  = 477d469dec0b8762
w0 = 84be85ce9804e94b    k0 = ec2802d4e0a488e9
```

Using the S-Box σ_0.

```
QARMA64_5
C = de5b629abbfcde27

QARMA64_6
C = dd117a6e8a55b338

QARMA64_7
C = 6faabc460bc8a784
```

Using the S-Box σ_1.

```
QARMA64_5
C = eb464d057108eb0e

QARMA64_6
C = 18c7547b7af08e42

QARMA64_7
C = 376f170678c80c7c
```

Using the S-Box σ_2.

```
QARMA64_5
C = d9f84b6d6b06c7c1

QARMA64_6
C = b6fbe247dd031812

QARMA64_7
C = bac6be9ee3d9e519
```

A.4 QARMA-128 with $M = M_{8,4}$ and $Q = Q_{8,4}$ (legacy)

```
P  = 2fdbb6a2c395e959 fdfa964e98c1a2e7    T  = 242767fd3486a96d fe9c904be82756a2
w0 = 58948b7ef8e5ec7e f9f8f014de0924eb    k0 = 4e7ce2081002eda4 fe20aaa4d868be09
```

Using the S-Box σ_0.

```
QARMA128_9
P = 2fdbb6a2c395e959 fdfa964e98c1a2e7
C = 4f9945f78bc95fdf 2b7ef9eb85faa032

QARMA128_10
P = 2fdbb6a2c395e959 fdfa964e98c1a2e7
C = be8343304ec23a37 7c73e199b0e37cf8

QARMA128_11
P = 2fdbb6a2c395e959 fdfa964e98c1a2e7
C = f96dffce5a7c0ece 2eff222f71e081d0
```

Using the S-Box σ_1.

```
QARMA128_9
P = 2fdbb6a2c395e959 fdfa964e98c1a2e7
C = c68d7fb1e329830c d29b626efa4015f0

QARMA128_10
P = 2fdbb6a2c395e959 fdfa964e98c1a2e7
C = c155ffbbac8505e8 790f72a36088dec5

QARMA128_11
P = 2fdbb6a2c395e959 fdfa964e98c1a2e7
C = f255b96cd7eeef39 28cbfcf3b5b4b191
```

Using the S-Box σ_2.

```
QARMA128_9
P = 2fdbb6a2c395e959 fdfa964e98c1a2e7
C = 7564cddcf60ecc04 bae52c17e31140ca

QARMA128_10
P = 2fdbb6a2c395e959 fdfa964e98c1a2e7
C = b52d6eac88dc5c76 bea485559b7d20d5

QARMA128_11
P = 2fdbb6a2c395e959 fdfa964e98c1a2e7
C = 0c310275414c10e9 4364623c9efa7b36
```

B Alternative Tweak Masking Function

We describe now a different candidate for modifying the tweak using a key, specially designed for 64-bit values, the *scrambler*. Depicted in Figure 12, it follows the Kam and Davida methodology to design *tree-structured substitution-permutation networks* [KD79]. Kam and Davida provide a concrete construction method, by means of which, a complete SPN with block size of $n = m^g$ bits can be built using g substitution layers, constructed from m-bit S-Boxes, and $g-1$ bit permutation layers. Our parameters are $(m, g) = (4, 3)$.

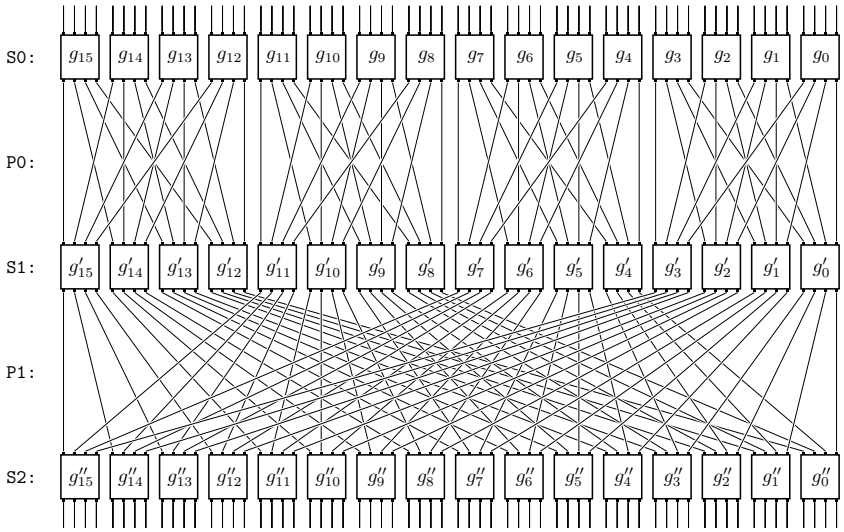

Figure 12: The Kam-Davida Three-Round 64-bit Scrambler

In order for each input bit to the scrambler to influence each output bit of the scrambler non-linearly, it is necessary that each input bit of the S-Box influences each output bit of the S-Box non-linearly (cf. § 3.3.2). This also means that in order to obtain full diffusion to the output the S-Box σ_0 should not be used, but σ_1 and σ_2 are good choices.

With respect to Figure 12, the scrambler's layers are designed as follows:

- At the beginning, IS is an address or a nonce t.
- If the internal state is represented as IS[63..0], the functions g_i, g'_i, g''_i update the i-th 4-bit segment IS$[4\,i+3\,..\,4\,i]$ of IS, as follows:
  - $g_i(\text{IS}[4\,i+3\,..\,4\,i]) = \sigma(\text{IS}[4\,i+3\,..\,4\,i] + \beta[4\,i+3\,..\,4\,i])$;
  - $g'_i(\text{IS}[4\,i+3\,..\,4\,i]) = \sigma(\text{IS}[4\,i+3\,..\,4\,i] + (t \ggg 32)[4\,i+3\,..\,4\,i])$; and

– $g_i''(\mathrm{IS}[4\,i+3\,..\,4\,i]) = \sigma(\mathrm{IS}[4\,i+3\,..\,4\,i] + \beta[4\,i+3\,..\,4\,i])$.

- The permutation layer `P0` maps the bit with index $16\,a + 4\,b + c$ of the state to the bit with index $16\,a + 4\,c + b$ for $0 \le a, b, c \le 3$.
- The permutation layer `P1` maps the bit with index $16\,a + b$ to the bit with index $4\,b + a$ for $0 \le a \le 3$, $0 \le b \le 15$.

Cryptographically, when analysed as a large, 64-bit S-Box, the scrambler is sub-optimal. In fact in [WT85] it was shown that Kam-Davida networks *dampen* the avalanche property somewhat. The extent to which this may be relevant for our applications will have to be studied.

C Proof that $M_{8,2}$ has 2^{72} Fixed Points on the State

Proposition 2. *The matrix* $M = \mathrm{circ}(0, \rho, \rho^4, \rho^5)$ *acting on* $R^{4\times4}$ *where* R *is the ring* $R_8 = \mathbb{F}_2[\rho] = \mathbb{F}_2[X]/(X^8+1)$ *has* 2^{72} *fixed points.*

Proof. Since M acts on the four columns of $R^{4\times4}$ independently, it will suffice to establish that the number of fixed points of M acting on R^4, i.e. on columns, is 2^{18}. A vector $V = (v_0, v_1, v_2, v_3)^t$ is a fixed point of M if and only if

$$(M+I)\cdot V = \begin{pmatrix} v_0 + v_1\,\rho + v_2\,\rho^4 + v_3\,\rho^5 \\ v_0\,\rho^5 + v_1 + v_2\,\rho + v_3\,\rho^4 \\ v_0\,\rho^4 + v_1\,\rho^5 + v_2 + v_3\,\rho \\ v_0\,\rho + v_1\,\rho^4 + v_2\,\rho^5 + v_3 \end{pmatrix} = \begin{pmatrix} u_0 \\ u_1 \\ u_2 \\ u_3 \end{pmatrix}$$

is zero. Now, $0 = u_0 + v_1\,\rho$, we get $v_0(1+\rho^6) + v_2(\rho^2+\rho^4) = 0$ or, in other words

$$(1+\rho^2)(v_0 + v_2\,\rho^4) = 0$$

which means that v_0 is equal to $v_2\,\rho^4$ up to an element annihilated by $1+\rho^2$, i.e.

$$v_0 = v_2\,\rho^4 + (1+\rho^2+\rho^4+\rho^6)\,\zeta \ .$$

Similarly,

$$v_1 = v_3\,\rho^4 + (1+\rho^2+\rho^4+\rho^6)\,\zeta' \ .$$

Note that ζ and ζ' are effectively defined modulo $1+\rho^2$, and they can only define four distinct values of v_0 (resp. v_1) for each fixed value of v_2 (resp. v_3). Substituting these expressions for v_0 and v_1 in $u_0 = 0$ (or $u_2 = 0$) we get

$$(1+\rho^2+\rho^4+\rho^6)(\zeta + \rho\,\zeta') = 0 \ .$$

We see that for each value of ζ there is a unique ζ' (modulo $1+\rho^2$) such that the last equation is satisfied. Using u_1 (or u_3) in place of u_0 or u_2 leads to the same result. The elements in $\ker(M+I) \subseteq R^4$ are thus uniquely determined by two components and a single two bit value, resulting in a cardinality of 2^{18}. □