# PRESENT: An Ultra-Lightweight Block Cipher

A. Bogdanov[1], L.R. Knudsen[2], G. Leander[1], C. Paar[1], A. Poschmann[1],
M.J.B. Robshaw[3], Y. Seurin[3], and C. Vikkelsoe[2]

[1] Horst-Görtz-Institute for IT-Security, Ruhr-University Bochum, Germany
[2] Technical University Denmark, DK-2800 Kgs. Lyngby, Denmark
[3] France Telecom R&D, Issy les Moulineaux, France
leander@rub.de, {abogdanov,cpaar,poschmann}@crypto.rub.de
lars@ramkilde.com, chv@mat.dtu.dk
{matt.robshaw,yannick.seurin}@orange-ftgroup.com

**Abstract.** With the establishment of the AES the need for new block ciphers has been greatly diminished; for almost all block cipher applications the AES is an excellent and preferred choice. However, despite recent implementation advances, the AES is not suitable for extremely constrained environments such as RFID tags and sensor networks. In this paper we describe an ultra-lightweight block cipher, PRESENT. Both security and hardware efficiency have been equally important during the design of the cipher and at 1570 GE, the hardware requirements for PRESENT are competitive with today's leading compact stream ciphers.

## 1 Introduction

One defining trend of this century's IT landscape will be the extensive deployment of tiny computing devices. Not only will these devices feature routinely in consumer items, but they will form an integral part of a pervasive — and unseen — communication infrastructure. It is already recognized that such deployments bring a range of very particular security risks. Yet at the same time the cryptographic solutions, and particularly the cryptographic primitives, we have at hand are unsatisfactory for extremely resource-constrained environments.

In this paper we propose a new hardware-optimized block cipher that has been carefully designed with area and power constraints uppermost in our mind. Yet, at the same time, we have tried to avoid a compromise in security. In achieving this we have looked back at the pioneering work embodied in the DES [34] and complemented this with features from the AES finalist candidate Serpent [4] which demonstrated excellent performance in hardware.

At this point it would be reasonable to ask why we might want to design a new block cipher. After all, it has become an "accepted" fact that stream ciphers are, potentially, more compact. Indeed, renewed efforts to understand the design of compact stream ciphers are underway with the eSTREAM [15] project and several promising proposals offer appealing performance profiles. But we note a couple of reasons why we might want to consider a compact block cipher. First, a block cipher is a versatile primitive and by running a block cipher in *counter* *mode* (say) we get a stream cipher. But second, and perhaps more importantly, the art of block cipher design seems to be a little better understood than that of stream ciphers. For instance, while there is a rich theory under-pinning the use of linear feedback shift registers [29] it is not easy to combine these building blocks to give a secure proposal. We suspect that a carefully designed block cipher could be a less risky undertaking than a newly designed stream cipher. Thus, we feel that a block cipher that requires similar hardware resources as a compact stream cipher could be of considerable interest.

It is important to realise that in developing a new block cipher, particularly one with aggressive performance characteristics, we are not just looking for innovative implementation. Rather, the design and implementation of the cipher go hand-in-hand and this has revealed several fundamental limits and inherent contradictions. For instance, a given security level places lower bounds on the block length and key length. Just processing a 64-bit state with an 80-bit key places fundamental lower limits on the amount of space we require. We also observe that hardware implementation — particularly compact hardware implementation — favours repetition. Even minor variations can have an unfortunate effect on the space required for an implementation. Yet, at the same time, the cryptanalyst also favours repetition and seeks mathematical structures that propagate easily across many rounds. How much simple, repetitive structure can we include without compromising its security?

In this paper we describe the compact block cipher[1] PRESENT. After a brief survey of the existing literature, the rest of the paper is organised in a standard way. PRESENT is described in Section 3 with the design decisions described in Section 4. The security analysis follows in Section 5 along with a detailed performance analysis in Section 6. We close the paper with our conclusions.

## 2 Existing Work

While there is a growing body of work on low-cost cryptography, the number of papers dealing with ultra-lightweight ciphers is surprisingly limited. Since our focus is on algorithm design we won't refer to work on low-cost communication and authentication protocols. Some of the most extensive work on compact implementation is currently taking place within the eSTREAM project. As part of that initiative, new stream ciphers suitable for efficient hardware implementation have been proposed. While this work is ongoing, some promising candidates are emerging [7, 19]. While the trade-offs are complex, implementation papers [18] suggest that around 1300-2600 *gate equivalents* (GE) would be required for the more compact ciphers within the eSTREAM project.

With regards to block ciphers it is well-known that DES was designed with hardware efficiency in mind. Given the very limited state of semiconductor circuits in the early 1970s, it is not surprising that DES possesses very competitive implementation properties. Work on DES reveals an implementaton of around

[1] The name reflects its similarity to Serpent and the goal of fitting everywhere; the very nature of ubiquitous computing.

generateRoundKeys()
**for** $i = 1$ to 31 **do**
  addRoundKey(STATE,$K_i$)
  sBoxLayer(STATE)
  pLayer(STATE)
**end for**
addRoundKey(STATE,$K_{32}$)

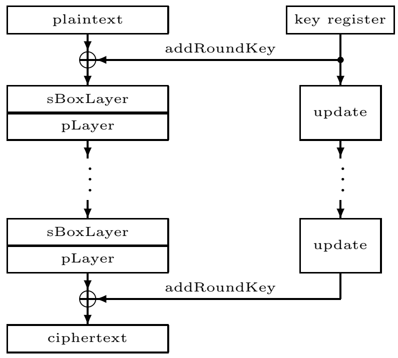


**Fig. 1.** A top-level algorithmic description of PRESENT.

3000 GE [42] while a serialized implementation can be realized with around 2300 GE [37]. The key length of DES limits its usefulness in many applications and makes proposals such as DESXL (2168 GE) of some considerable interest [37].

For modern block ciphers, the landmark paper of [16] gives a very thorough analysis of a low-cost implementation of the AES [35]. However, the resources required for this cipher are around 3600 GE, which is an indirect consequence of the fact that Rijndael was designed for software efficiency on 8- and 32-bit processors. Implementation requirements for the *Tiny Encryption Algorithm* TEA [43, 44] are not known, but a crude estimate is that TEA needs at least 2100 GE and XTEA needs[2] at least 2000 GE. Four dedicated proposals for low-cost implementation are mCRYPTON [30], HIGHT [22], SEA [41], and CGEN [40], though the latter is not primarily intended as a block cipher. mCRYPTON has a precise hardware assessment and requires 2949 GE, HIGHT requires around 3000 GE while SEA with parameters comparable to PRESENT requires around 2280 GE.

## 3 The Block Cipher PRESENT

PRESENT is an example of an SP-network [33] and consists of 31 rounds. The block length is 64 bits and two key lengths of 80 and 128 bits are supported. Given the applications we have in mind, we recommend the version with 80-bit keys. This is more than adequate security for the low-security applications typically required in tag-based deployments, but just as importantly, this matches the design goals of hardware-oriented stream ciphers in the eSTREAM project and allows us to make a fairer comparison. The security claims and performance attributes of the 128-bit version are provided in an appendix.

[2] These figures and others in Section 2 are "back-of-an-envelope" where we assume the following requirements: 32-bit XOR = 80 GE, 32-bit arithmetic ADD = 148 GE, 192-bit FF = 1344 GE, SHIFT = 0 GE. All estimated figures lack any control logic which might significantly increase the required area.

Each of the 31 rounds consists of an XOR operation to introduce a round key $K_i$ for $1 \le i \le 32$, where $K_{32}$ is used for post-whitening, a linear bitwise permutation and a non-linear substitution layer. The non-linear layer uses a single 4-bit S-box $S$ which is applied 16 times in parallel in each round. The cipher is described in pseudo-code in Figure 1, and each stage is now specified in turn. The design rationale are given in Section 4 and throughout we number bits from zero with bit zero on the right of a block or word.

**addRoundKey.** Given round key $K_i = \kappa_{63}^i \dots \kappa_0^i$ for $1 \le i \le 32$ and current STATE $b_{63} \dots b_0$, addRoundKey consists of the operation for $0 \le j \le 63$,

$$b_j \to b_j \oplus \kappa_j^i.$$

**sBoxlayer.** The S-box used in PRESENT is a 4-bit to 4-bit S-box $S : \mathbb{F}_2^4 \to \mathbb{F}_2^4$. The action of this box in hexadecimal notation is given by the following table.

| $x$ | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | A | B | C | D | E | F |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $S[x]$ | C | 5 | 6 | B | 9 | 0 | A | D | 3 | E | F | 8 | 4 | 7 | 1 | 2 |

For sBoxLayer the current STATE $b_{63} \dots b_0$ is considered as sixteen 4-bit words $w_{15} \dots w_0$ where $w_i = b_{4*i+3}||b_{4*i+2}||b_{4*i+1}||b_{4*i}$ for $0 \le i \le 15$ and the output nibble $S[w_i]$ provides the updated state values in the obvious way.

**pLayer.** The bit permutation used in PRESENT is given by the following table. Bit $i$ of STATE is moved to bit position $P(i)$.

| $i$ | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $P(i)$ | 0 | 16 | 32 | 48 | 1 | 17 | 33 | 49 | 2 | 18 | 34 | 50 | 3 | 19 | 35 | 51 |
| $i$ | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30 | 31 |
| $P(i)$ | 4 | 20 | 36 | 52 | 5 | 21 | 37 | 53 | 6 | 22 | 38 | 54 | 7 | 23 | 39 | 55 |
| $i$ | 32 | 33 | 34 | 35 | 36 | 37 | 38 | 39 | 40 | 41 | 42 | 43 | 44 | 45 | 46 | 47 |
| $P(i)$ | 8 | 24 | 40 | 56 | 9 | 25 | 41 | 57 | 10 | 26 | 42 | 58 | 11 | 27 | 43 | 59 |
| $i$ | 48 | 49 | 50 | 51 | 52 | 53 | 54 | 55 | 56 | 57 | 58 | 59 | 60 | 61 | 62 | 63 |
| $P(i)$ | 12 | 28 | 44 | 60 | 13 | 29 | 45 | 61 | 14 | 30 | 46 | 62 | 15 | 31 | 47 | 63 |

**The key schedule.** PRESENT can take keys of either 80 or 128 bits. However we focus on the version with 80-bit keys. The user-supplied key is stored in a key register $K$ and represented as $k_{79}k_{78} \dots k_0$. At round $i$ the 64-bit round key $K_i = \kappa_{63}\kappa_{62} \dots \kappa_0$ consists of the 64 leftmost bits of the current contents of register $K$. Thus at round $i$ we have that:

$$K_i = \kappa_{63}\kappa_{62} \dots \kappa_0 = k_{79}k_{78} \dots k_{16}.$$

After extracting the round key $K_i$, the key register $K = k_{79}k_{78} \dots k_0$ is updated as follows.

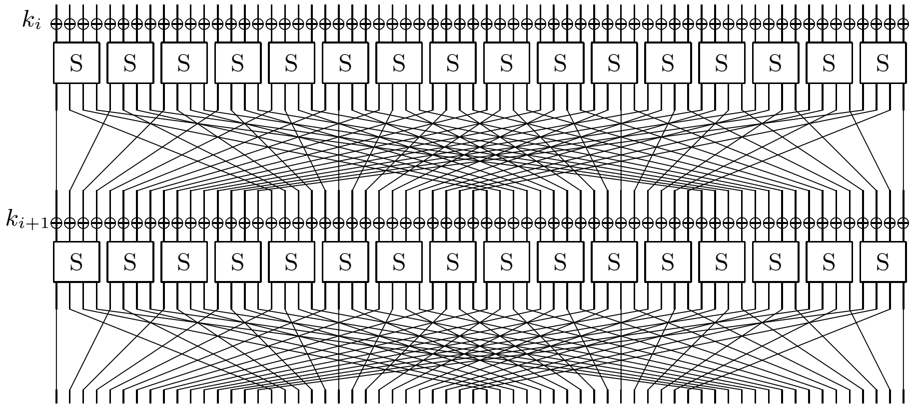

**Fig. 2.** . The S/P network for PRESENT.

1. $[\mathtt{k}_{79}\mathtt{k}_{78}\ldots\mathtt{k}_{1}\mathtt{k}_{0}] = [\mathtt{k}_{18}\mathtt{k}_{17}\ldots\mathtt{k}_{20}\mathtt{k}_{19}]$
2. $[\mathtt{k}_{79}\mathtt{k}_{78}\mathtt{k}_{77}\mathtt{k}_{76}] = S[\mathtt{k}_{79}\mathtt{k}_{78}\mathtt{k}_{77}\mathtt{k}_{76}]$
3. $[\mathtt{k}_{19}\mathtt{k}_{18}\mathtt{k}_{17}\mathtt{k}_{16}\mathtt{k}_{15}] = [\mathtt{k}_{19}\mathtt{k}_{18}\mathtt{k}_{17}\mathtt{k}_{16}\mathtt{k}_{15}] \oplus$ `round_counter`

Thus, the key register is rotated by 61 bit positions to the left, the left-most four bits are passed through the PRESENT S-box, and the `round_counter` value $i$ is exclusive-ored with bits $\mathtt{k}_{19}\mathtt{k}_{18}\mathtt{k}_{17}\mathtt{k}_{16}\mathtt{k}_{15}$ of $K$ with the least significant bit of `round_counter` on the right. The key schedule for 128-bit keys is presented in an appendix.

## 4 Design Issues for PRESENT

Besides security and efficient implementation, the main goal when designing PRESENT was simplicity. It is therefore not surprising that similar designs have been considered in other contexts [21] and can even be used as a tutorial for students [20]. In this section we justify the decisions we took during the design of PRESENT. First, however, we describe the anticipated application requirements.

### 4.1 Goals and environment of use

In designing a block cipher suitable for extremely constrained environments, it is important to recognise that we are not building a block cipher that is necessarily suitable for wide-spread use; we already have the AES [35] for this. Instead, we are targeting some very specific applications for which the AES is unsuitable. These will generally conform to the following characteristics.

– The cipher is to be implemented in hardware.

– Applications will only require moderate security levels. Consequently, 80-bit security will be adequate. Note that this is also the position taken for hardware profile stream ciphers submitted to eSTREAM [15].
– Applications are unlikely to require the encryption of large amounts of data. Implementations might therefore be optimised for performance or for space without too much practical impact.
– In some applications it is possible that the key will be fixed at the time of device manufacture. In such cases there would be no need to re-key a device (which would incidentally rule out a range of key manipulation attacks).
– After security, the physical space required for an implementation will be the primary consideration. This is closely followed by peak and average power consumption, with the timing requirements being a third important metric.
– In applications that demand the most efficient use of space, the block cipher will often only be implemented as *encryption-only*. In this way it can be used within challenge-response authentication protocols and, with some careful state management, it could be used for both encryption and decryption of communications to and from the device by using the counter mode [36].

Taking such considerations into account we decided to make PRESENT a 64-bit block cipher with an 80-bit key[3]. Encryption and decryption with PRESENT have roughly the same physical requirements. Opting to support both encryption and decryption will result in a lightweight block cipher implementation that is still smaller than an encryption-only AES. Opting to implement an encryption-only PRESENT will give an ultra-lightweight solution. The encryption subkeys can be computed *on-the-fly*.

The literature contains a range of attacks that manipulate time-memory-data trade-offs [6] or the birthday paradox when encrypting large amounts of data. However such attacks depend solely on the parameters of the block cipher and exploit no inner structure. Our goal is that these attacks be the best available to an adversary. Side-channel and invasive hardware attacks are likely to be a threat to PRESENT, as they are to all cryptographic primitives. For the likely applications, however, the moderate security requirements reflect the very limited gain any attacker would make in practice. In a risk assessment, such attacks are unlikely to be a significant factor.

### 4.2 The permutation layer

When choosing the mixing layer, our focus on hardware efficiency demands a linear layer that can be implemented with a minimum number of processing elements, *i.e.* transistors. This leads us directly to bit permutations. Given our focus on simplicity, we have chosen a regular bit-permutation and this helps to make a clear security analysis (see Section 5).

[3] Appendix II gives an option for 128-bit keys but we do not expect it to be used.

### 4.3 The S-box.

We use a single 4-bit to 4-bit S-box $S : \mathbb{F}_2^4 \to \mathbb{F}_2^4$ in PRESENT. This is a direct consequence of our pursuit of hardware efficiency, with the implementation of such an S-box typically being much more compact than that of an 8-bit S-box. Since we use a bit permutation for the linear diffusion layer, AES-like diffusion techniques [12] are not an option for PRESENT. Therefore we place some additional conditions on the S-boxes to improve the so-called *avalanche of change*. More precisely, the S-box for PRESENT fullfils the following conditions, where we denote the Fourier coefficient of $S$ by

$$S_b^W(a) = \sum_{x \in \mathbb{F}_2^4} (-1)^{\langle b, S(x) \rangle + \langle a, x \rangle}.$$

1. For any fixed non-zero input difference $\Delta_I \in \mathbb{F}_2^4$ and any fixed non-zero output difference $\Delta_O \in \mathbb{F}_2^4$ we require
$$\#\{x \in \mathbb{F}_2^4 \,|\, S(x) + S(x + \Delta_I) = \Delta_O\} \leq 4.$$
2. For any fixed non-zero input difference $\Delta_I \in \mathbb{F}_2^4$ and any fixed output difference $\Delta_O \in \mathbb{F}_2^4$ such that $\mathrm{wt}(\Delta_I) = \mathrm{wt}(\Delta_O) = 1$ we have
$$\{x \in \mathbb{F}_2^4 \,|\, S(x) + S(x + \Delta_I) = \Delta_O\} = \emptyset.$$
3. For all non-zero $a \in \mathbb{F}_2^4$ and all non-zero $b \in \mathbb{F}_4$ it holds that $|S_b^W(a)| \leq 8$.
4. For all $a \in \mathbb{F}_2^4$ and all non-zero $b \in \mathbb{F}_4$ such that $\mathrm{wt}(a) = \mathrm{wt}(b) = 1$ it holds that $S_b^W(a) = \pm 4$.

As will become clear in Section 5, these conditions will ensure that PRESENT is resistant to differential and linear attacks. Using a classification of all 4-bit S-boxes that fulfill the above conditions [27] we chose an S-box that is particular well-suited to efficient hardware implementation.

## 5 Security Analysis

We now present the results of a security analysis of PRESENT.

### 5.1 Differential and linear cryptanalysis

Differential [3] and linear [32] cryptanalysis are among the most powerful techniques available to the cryptanalyst. In order to gauge the resistance of PRESENT to differential and linear cryptanalysis we provide a lower bound to the number of so-called *active* S-boxes involved in a differential (or linear) characteristic.

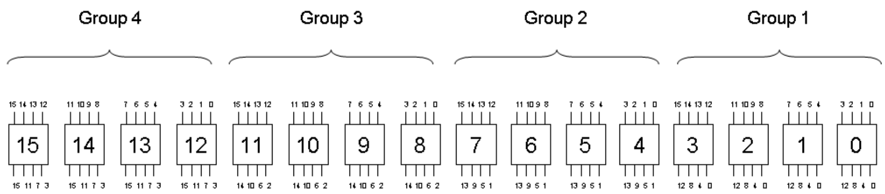


**Fig. 3.** The grouping of S-boxes in PRESENT for the purposes of cryptanalysis. The input numbers indicate the S-box origin from the preceeding round and the output numbers indicate the destination S-box in the following round.

**Differential cryptanalysis.** The case of differential cryptanalysis is captured by the following theorem.

**Theorem 1.** *Any five-round differential characteristic of* PRESENT *has a minimum of 10 active S-boxes.*

While Theorem 1 will be formally proved in Appendix III, we make the following observations. We divide the 16 S-boxes into four groups (see Figure 3) and by examining the permutation layer one can then establish the following.

1. The input bits to an S-box come from 4 distinct S-boxes of the same group.
2. The input bits to a group of four S-boxes come from 16 different S-boxes.
3. The four output bits from a particular S-box enter four distinct S-boxes, each of which belongs to a distinct group of S-boxes in the subsequent round.
4. The output bits of S-boxes in distinct groups go to distinct S-boxes.

The proof of Theorem 1 in Appendix III follows from these observations. By using Theorem 1 any differential characteristic over 25 rounds of PRESENT must have at least $5 \times 10 = 50$ active S-boxes. The maximum differential probability of a PRESENT S-box is $2^{-2}$ and so the probability of a single 25-round differential characteristic is bounded by $2^{-100}$. Advanced techniques allow the cryptanalyst to remove the outer rounds from a cipher to exploit a shorter characteristic. However even if we allow an attacker to remove six rounds from the cipher, a situation without precedent, then the data required to exploit the remaining 25-round differential characteristic exceeds the amount available. Thus, the security bounds are more than we require. However, we have practically confirmed that the bound on the number of active S-boxes in Theorem 1 is tight.

**Practical confirmation.** We can identify characteristics that involve ten S-boxes over five rounds. The following two-round iterative characteristic involves two S-boxes per round and holds with probability $2^{-25}$ over five rounds.

$$\begin{aligned} \Delta &= \texttt{0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1} \\ &\rightarrow \texttt{0 0 0 0 0 0 0 0 0 0 0 3 0 0 0 3} \\ &\rightarrow \texttt{0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1} = \Delta. \end{aligned}$$

A more complicated characteristic holds with probability $2^{-21}$ over five rounds.

$$\begin{aligned}\Delta &= \texttt{0 0 0 0 0 0 0 0 0 0 0 0 7 0 7 0}\\ &\rightarrow \texttt{0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 A}\\ &\rightarrow \texttt{0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0}\\ &\rightarrow \texttt{0 0 0 0 0 0 0 0 1 0 0 0 1 0 0 0}\\ &\rightarrow \texttt{0 0 0 0 0 0 0 0 0 0 8 8 0 0 8 8}\\ &\rightarrow \texttt{0 0 3 3 0 0 0 0 0 0 3 3 0 0 3 3}.\end{aligned}$$

While the probability of this second characteristic is very close to the bound of $2^{-20}$, it is non-iterative and of little practical value. Instead we have experimentally confirmed the probability of the two-round iterative differential. In experiments over 100 independent sub-keys using $2^{23}$ chosen plaintext pairs, the observed probability was as predicted. This seems to suggest that for this particular characteristic there is no accompanying significant differential. However, determining the extent of any differential effect is a complex and time-consuming task even though our preliminary analysis has been encouraging.

**Linear cryptanalysis.** The case of the linear cryptanalysis of PRESENT is handled by the following theorem where we analyse the best linear approximation to four rounds of PRESENT.

**Theorem 2.** *Let $\epsilon_{4\mathrm{R}}$ be the maximal bias of a linear approximation of four rounds of* PRESENT*. Then $\epsilon_{4\mathrm{R}} \leq \frac{1}{2^7}$.*

The theorem is formally proved in Appendix IV, and we can use it directly to bound the maximal bias of a 28-round linear approximation by

$$2^6 \times \epsilon_{4\mathrm{R}}^7 = 2^6 \times (2^{-7})^7 = 2^{-43}.$$

Therefore under the assumption that a cryptanalyst need only approximate 28 of the 31 rounds in PRESENT to mount a key recovery attack, linear cryptanalysis of the cipher would require of the order of $2^{84}$ known plaintext/ciphertexts. Such data requirements exceed the available text.

**Some advanced differential/linear attacks.** The structure of PRESENT allows us to consider some dedicated forms of attacks. However none have yielded an attack that requires less text than the lower bound on text requirements for linear cryptanalysis. Among the dedicated attacks we considered was one using palindromic differences, since symmetrical differences are preserved with probability one over the diffusion layer, and some advanced variants of differential-linear attacks [28]. While the attacks seemed promising over a few rounds, they very quickly lost their practical value and are unlikely to be useful in the cryptanalysis of PRESENT. We also established that *truncated differential cryptanalysis* [23, 24] was likely to have limited value, though the following two-round truncated extension holds with probability one.

$$
\begin{aligned}
\Delta = {} & \mathtt{0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,1\,1} \\
& \rightarrow \mathtt{0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,3\,0\,0\,0\,3} \text{ [ iterate the two-round characteristic ]} \\
& \rightarrow \qquad \vdots \\
& \rightarrow \mathtt{0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,1\,1} \\
& \rightarrow \mathtt{0\,0\,0\,?\,0\,0\,0\,?\,0\,0\,0\,?\,0\,0\,0\,3} \\
& \rightarrow \delta_0\,\delta_1\,\delta_2\,\delta_3\,\delta_4\,\delta_5\,\delta_6\,\delta_7\,\delta_8\,\delta_9\,\delta_{10}\,\delta_{11}\,\delta_{12}\,\delta_{13}\,\delta_{14}\,\delta_{15} \quad \text{where all } \delta_i \in \{0,1\}.
\end{aligned}
$$

Even when used to reduce the length of the differential characteristics already identified, the data requirements still remain excessive.

### 5.2 Structural attacks

Structural attacks such as *integral attacks* [25] and *bottleneck attacks* [17] are well-suited to the analysis of AES-like ciphers [12, 13, 38]. Such ciphers have strong word-like structures, where the words are typically bytes. However the design of PRESENT is almost exclusively bitwise, and while the permutation operation is somewhat regular, the development and propagation of word-wise structures are disrupted by the bitwise operations used in the cipher.

### 5.3 Algebraic attacks

Algebraic attacks have had better success when applied to stream ciphers than block ciphers. Nevertheless, the simple structure of PRESENT means that they merit serious study. The PRESENT S-box is described by 21 quadratic equations in the eight input/output-bit variables over $GF(2)$. This is not surprising since it is well-known that any four bit S-box can be described by at least 21 such equations. The entire cipher can then described by $e = n \times 21$ quadratic equations in $v = n \times 8$ variables, where $n$ is the number of S-boxes in the encryption algorithm and the key schedule. For PRESENT we have $n = (31 \times 16) + 31$ thus the entire system consists of $11,067$ quadratic equations in $4,216$ variables.

The general problem of solving a system of multivariate quadratic equations is NP-hard. However the systems derived for block ciphers are very sparse since they are composed of $n$ small systems connected by simple linear layers. Nevertheless, it is unclear whether this fact can be exploited in a so-called algebraic attack. Some specialised techniques such as XL [10] and XSL [11] have been proposed, though flaws in both techniques have been discovered [8, 14]. Instead the only practical results on the algebraic cryptanalysis of block ciphers have been obtained by applying the Buchberger and F4 algorithms within Magma [31]. Simulations on small-scale versions of the AES showed that for all but the very smallest SP-networks one quickly encounters difficulties in both time and memory complexity [9]. The same applies to PRESENT.

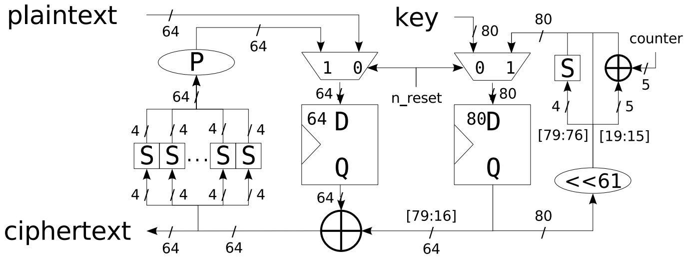


**Fig. 4.** The datapath of an area-optimized version of PRESENT-80.

**Practical confirmation.** We ran simulations on small-scale versions using the $F_4$ algorithm in Magma. When there is a single S-box, *i.e.* a very small block size of four bits, then Magma can solve the resulting system of equations over many rounds. However, by increasing the block size and adding S-boxes, along with an appropriate version of the linear diffusion layer, the system of equations soon becomes too large. Even when considering a system consisting of seven S-boxes, *i.e.* a block size of 28 bits, we were unable to get a solution in a reasonable time to a two-round version of the reduced cipher. Our analysis suggests that algebraic attacks are unlikely to pose a threat to PRESENT.

### 5.4 Key schedule attacks

Since there are no established guidelines to the design of key schedules, there is both a wide variety of designs and a wide variety of schedule-specific attacks. The most effective attacks come under the general heading of *related-key attacks* [2] and *slide attacks* [5], and both rely on the build-up of identifiable relationships between different sets of subkeys. To counter this threat, we use a round-dependent counter so that subkey sets cannot easily be "slid", and we use a non-linear operation to mix the contents of the key register $K$. In particular,

- all bits in the key register are a non-linear function of the 80-bit user-supplied key by round 21,
- that each bit in the key register after round 21 depends on at least four of the user-supplied key bits, and
- by the time we arrive at deriving $K_{32}$, six bits are degree two expressions of the 80 user-supplied key bits, 24 bits are of degree three, while the remaining bits are degree six or degree nine function of the user-supplied key bits.

We believe these properties to be sufficient to resist key schedule-based attacks.

## 6 Hardware performance

We implemented PRESENT-80 in VHDL and synthesized it for the Virtual Silicon (VST) standard cell library based on the UMC L180 $0.18\mu$ 1P6M Logic process. We used *Mentor Graphics Modelsim SE PLUS 5.8c* for simulation and *Synopsys Design Compiler* version *Y-2006.06* for synthesis and power simulation. The foundry typical values (of 1.8 Volt for the core voltage and 25°C for the temperature) were used and the suggested wireload model was applied for the power simulation. Note that this is suitable for designs around 10,000 GE so the power results will be pessimistic for significantly smaller designs. Figure 4 shows the datapath of an area-optimized encryption-only PRESENT-80, which performs one round in one clock cycle *i.e.* a 64-bit width datapath. Note that during the design phase of PRESENT we use the same S-box 16 times rather than having 16 different S-boxes and this eases a further serialization of the design, *i.e.* with a 4-bit width datapath. Our implementation requires 32 clock cycles to encrypt a 64-bit plaintext with an 80-bit key, occupies 1570 GE and has a simulated power consumption of 5$\mu$W.

| module | GE | % | module | GE | % |
|---|---|---|---|---|---|
| data state | 384.39 | 24.48 | KS: key state | 480.49 | 30.61 |
| s-layer | 448.45 | 28.57 | KS: S-box | 28.03 | 1.79 |
| p-layer | 0 | 0 | KS: Rotation | 0 | 0 |
| counter: state | 28.36 | 1.81 | KS: counter-XOR | 13.35 | 0.85 |
| counter: combinatorial | 12.35 | 0.79 | key-XOR | 170.84 | 10.88 |
| other | 3.67 | 0.23 | | | |
| | | | **sum** | **1569.93** | **100** |

**Table 1.** Area requirement of PRESENT

The bulk of the area is occupied by flip-flops for storing the key and the data state, followed by the S-layer and the key-XOR. Bit permutations are simple wiring and will increase the area only when the implementation is taken to the place&route-step. Note that the main goal of our implementation was a small footprint in hardware, however, we also synthesized a power-optimized implementation. For an additional 53 GE we attain a power consumption of only 3.3$\mu$W and PRESENT-128 would occupy an estimated area of 1886 GE. Beside a very small footprint PRESENT has a rather high throughput giving good energy-per-bit. A comparison with other ciphers follows in Table 2.

## 7 Conclusions

In this paper we have described the new block cipher PRESENT. Our goal has been an ultra-lightweight cipher that offers a level of security commensurate with a 64-bit block size and an 80-bit key. Intriguingly PRESENT has implementation requirements similar to many compact stream ciphers. As such, we believe it to be of both theoretical and practical interest. Like all new proposals, we discourage the immediate deployment of PRESENT but strongly encourage its analysis.

| | Key size | Block size | Cycles per block | Throughput at 100KHz (Kbps) | Logic process | Area GE | Area rel. |
|---|---|---|---|---|---|---|---|
| **Block ciphers** | | | | | | | |
| PRESENT-80 | 80 | 64 | 32 | 200 | 0.18$\mu$m | 1570 | 1 |
| AES-128 [16] | 128 | 128 | 1032 | 12.4 | 0.35$\mu$m | 3400 | 2.17 |
| HIGHT [22] | 128 | 64 | 1 | 6400 | 0.25$\mu$m | 3048 | 1.65 |
| mCrypton [30] | 96 | 64 | 13 | 492.3 | 0.13$\mu$m | 2681 | 1.71 |
| Camellia [1] | 128 | 128 | 20 | 640 | 0.35$\mu$m | 11350 | 7.23 |
| DES [37] | 56 | 64 | 144 | 44.4 | 0.18$\mu$m | 2309 | 1.47 |
| DESXL [37] | 184 | 64 | 144 | 44.4 | 0.18$\mu$m | 2168 | 1.38 |
| **Stream ciphers** | | | | | | | |
| Trivium [18] | 80 | 1 | 1 | 100 | 0.13$\mu$m | 2599 | 1.66 |
| Grain [18] | 80 | 1 | 1 | 100 | 0.13$\mu$m | 1294 | 0.82 |

**Table 2.** Comparison of lightweight cipher implementations

## Acknowledgement

The work presented in this paper was supported in part by the European Commission within the STREP UbiSec&Sens of the EU Framework Programme 6 for Research and Development (www.ist-ubisecsens.org). The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the UbiSec&Sens project or the European Commission.

## References

1. K. Aoki, T. Ichikawa, M. Kanda, M. Matsui, S. Moriai, J. Nakajima, and T. Tokita. Camellia: A 128-Bit Block Cipher Suitable for Multiple Platforms - Design and Analysis. In D. Stinson and S. Tavares, editors, *Proceedings of SAC 2000*, pages 39–56, Springer-Verlag, 2000.
2. E. Biham. New Types of Cryptanalytic Attacks Using Related Keys. In T. Helleseth, editor, *Proceedings of Eurocrypt '93*, LNCS, volume 765, pages 398–409, Springer-Verlag, 1994.
3. E. Biham and A. Shamir. *Differential Cryptanalysis of the Data Encryption Standard*. Springer Verlag, 1993.
4. E. Biham, L.R. Knudsen, and R.J. Anderson. Serpent: A New Block Cipher Proposal. In S. Vaudenay, editor, *Proceedings of FSE 1998*, LNCS, volume 1372, pages 222–238, Springer Verlag.
5. A. Biryukov and D. Wagner. Advanced Slide Attacks. In B. Preneel, editor, *Proceedings of Eurocrypt 2000*, LNCS, volume 1807, pages 589–606, Springer-Verlag, 2000.
6. A. Biryukov, S. Mukhopadhyay, and P. Sarkar. Improved Time-memory Trade-offs with Multiple Data. In B. Preneel and S. Tavares, editors, *Proceedings of SAC 2005*, LNCS, volume 3897, pages 110-127, Springer Verlag.
7. C. de Cannière and B. Preneel. Trivium. Available via `www.ecrypt.eu.org`.

8. C. Cid and G. Leurent. An Analysis of the XSL Algorithm. In B. Roy, editor, *Proceedings of Asiacrypt 2005*, LNCS, volume 3788, pages 333–352, Springer-Verlag, 2005.
9. C. Cid, S. Murphy, and M.J.B. Robshaw. Small Scale Variants of the AES. In H. Gilbert and H. Handschuh, editors, *Proceedings of FSE 2005*, LNCS, volume 3557, pages 145–162, Springer-Verlag, 2005.
10. N. Courtois, A. Klimov, J. Patarin, and A. Shamir. Efficient Algorithms for Solving Overdefined Systems of Multivariate Polynomial Equations. In B. Preneel, editor, *Proceedings of Eurocrypt 2000*, LNCS, volume 1807, pages 392–407, Springer-Verlag, 2000.
11. N. Courtois and J. Pieprzyk. Cryptanalysis of Block Ciphers with Overdefined Systems of Equations. In Y. Zheng, editor, *Proceedings of Asiacrypt 2002*, LNCS, volume 2501, pages 267–287, Springer-Verlag, 2002.
12. J. Daemen and V. Rijmen. The Design of Rijndael. Springer-Verlag, 2002.
13. J. Daemen, L.R. Knudsen, and V. Rijmen. The Block Cipher Square. In E. Biham, editor, *Proceedings of FSE 1997*, LNCS, volume 1267, pages 149–165, Springer-Verlag, 2005.
14. C. Diem. The XL-Algorithm and a Conjecture from Commutative Algebra. In P.J. Lee, editor, *Proceedings of Asiacrypt 2004*, LNCS, volume 3329, pages 323–337, Springer-Verlag, 2004.
15. ECRYPT Network of Excellence. The Stream Cipher Project: eSTREAM. Available via `www.ecrypt.eu.org/stream`.
16. M. Feldhofer, S. Dominikus, and J. Wolkerstorfer. Strong Authentication for RFID Systems Using the AES algorithm. In M. Joye and J.-J. Quisquater, editors, *Proceedings of CHES 2004*, LNCS, volume 3156, pages 357–370, Springer Verlag, 2004.
17. H. Gilbert and M. Minier. A Collision Attack on 7 Rounds of Rijndael. In *Proceedings of Third Advanced Encryption Standard Conference*, National Institute of Standards and Technology, 230–241, 2000.
18. T. Good, W. Chelton, and M. Benaissa. *Hardware Results for Selected Stream Cipher Candidates*. Presented at SASC 2007, February 2007. Available for download via http://www.ecrypt.eu.org/stream/,
19. M. Hell, T. Johansson and W. Meier. Grain - A Stream Cipher for Constrained Environments. Available via `www.ecrypt.eu.org`.
20. H. Heys. A Tutorial on Differential and Linear Cryptanalysis. Available via `www.engr.mun.ca/~howard/PAPERS/ldc_tutorial.pdf`.
21. H. Heys and S. Tavares. Substitution-Permutation Networks Resistant to Differential and Linear Cryptanalysis. *Journal of Cryptology*, vol.9, no.1, pages 1–21, 1996.
22. D. Hong, J. Sung, S. Hong, J. Lim, S. Lee, B.-S; Koo, C. Lee, D. Chang, J. Lee, K. Jeong, H. Kim, J. Kim, and S. Chee. HIGHT: A New Block Cipher Suitable for Low-Resource Device. In L. Goubin and M. Matsui, editors, *Proceedings of CHES 2006*, LNCS, volume 4249, pages 46–59, Springer-Verlag, 2006.
23. L.R. Knudsen and T. Berson. Truncated Differentials of SAFER. In D. Gollman, editor, *Proceedings of FSE 1996*, LNCS, volume 1039, pages 15–26, Springer-Verlag, 1996.
24. L.R. Knudsen, M.J.B. Robshaw, and D. Wagner. Truncated Differentials and Skipjack. In M. Weiner, editor, *Proceedings of Crypto 99*, LNCS, volume 1666, pages 165–180, Springer-Verlag, 1999.
25. L.R. Knudsen and D. Wagner. Integral Cryptanalysis. In J. Daemen and V. Rijmen, editors, *Proceedings of FSE 2002*, LNCS, volume 2365, pages 112–127, Springer-Verlag, 2002.

26. X. Lai, J. Massey, and S. Murphy. Markov Ciphers and Differential Cryptanalysis. In D.W. Davies, editors, *Proceedings of Eurocrypt 91*, LNCS, volume 547, pages 17–38, Springer-Verlag, 1991.
27. G. Leander and A. Poschmann. On the Classification of 4 Bit S-boxes. In C. Carlet and B. Sunar, editors, *Proceedings of Arithmetic of Finite Fields, First International Workshop, WAIFI 2007*, LNCS , volume 4547, Springer, to appear.
28. M.E. Hellman and S.K. Langford. Differential-Linear Cryptanalysis. In Y. Desmedt, editors, *Proceedings of Crypto 94*, LNCS, volume 839, pages 17–25, Springer-Verlag, 1994.
29. R. Lidl and H. Niederreiter. Introduction to Finite Fields and their Applications. Revised edition. Cambridge University Press, 1994.
30. C. Lim and T. Korkishko. mCrypton - A Lightweight Block Cipher for Security of Low-cost RFID Tags and Sensors. In J. Song, T. Kwon, and M. Yung, editors, *Workshop on Information Security Applications - WISA'05*, LNCS, volume 3786, pages 243-258, Springer-Verlag, 2005.
31. MAGMA v2.12. Computational Algebra Group, School of Mathematics and Statistics, University of Sydney, 2005, http://magma.maths.usyd.edu.au.
32. M. Matsui. Linear Cryptanalysis Method for DES Cipher. In T. Helleseth, editor, *Proceedings of Eurocrypt '93*, LNCS, volume 765, pages 386–397, Springer-Verlag, 1994.
33. A. Menezes, P.C. van Oorschot, and S. Vanstone. The Handbook of Applied Cryptography. CRC Press, 1996.
34. National Institute of Standards and Technology. FIPS 46-3: Data Encryption Standard, March 1993. Available via `csrc.nist.gov`.
35. National Institute of Standards and Technology. FIPS 197: Advanced Encryption Standard, November 2001. Available via `csrc.nist.gov`.
36. National Institute of Standards and Technology. SP800-38A: Recommendation for block cipher modes of operation. December 2001. Available via `csrc.nist.gov`.
37. G. Leander, C Paar, A. Poschmann, and K Schramm A Family of Lightweight Block Ciphers Based on DES Suited for RFID Applications. In A. Biryukov, editor, *Proceedings of FSE 2007*, LNCS, Springer-Verlag, to appear.
38. V. Rijmen, J. Daemen, B. Preneel, A. Bosselaers, and E. De Win. The cipher Shark. In D. Gollman, editor, *Proceedings of FSE 1996*, LNCS, volume 1039, pages 99–112, Springer-Verlag, 1996.
39. R. Rivest. The RC5 Encryption Algorithm. In B. Preneel, editor, *Proceedings of FSE 1994*, LNCS, volume 1008, pages 363–366, Springer-Verlag, 1994.
40. M.J.B. Robshaw. Searching for compact algorithms: CGEN. In P.Q. Nguyen, editor, *Proceedings of Vietcrypt 2006*, LNCS, volume 4341, pages 37–49, Springer, 2006.
41. F.-X. Standaert, G. Piret, N. Gershenfeld, and J.-J. Quisquater. SEA: A Scalable Encryption Algorithm for Small Embedded Applications. In J. Domingo-Ferrer, J. Posegga, and D. Schreckling, editors, *Smart Card Research and Applications, Proceedings of CARDIS 2006*, LNCS, volume 3928, pages 222–236, Springer-Verlag.
42. I. Verbauwhede, F. Hoornaert, J. Vandewalle, and H. De Man, Security and Performance Optimization of a New DES Data Encryption Chip. *IEEE Journal of Solid-State Circuits*, vol.23, no.3, pages 647–656, 1988.
43. D. Wheeler and R. Needham. TEA, a Tiny Encryption Algorithm. In B. Preneel, editor, *Proceedings of FSE 1994*, LNCS, volume 1008, pages 363–366, Springer-Verlag, 1994.
44. D. Wheeler and R. Needham. TEA extensions. October, 1997. (Also Correction to XTEA. October, 1998.) Available via `www.ftp.cl.cam.ac.uk/ftp/users/djw3/`.

## Appendix I

Test vectors for PRESENT with an 80-bit key are shown in hexadecimal notation.

| *plaintext* | *key* | *ciphertext* |
|---|---|---|
| 00000000 00000000 | 00000000 00000000 0000 | 5579C138 7B228445 |
| 00000000 00000000 | FFFFFFFF FFFFFFFF FFFF | E72C46C0 F5945049 |
| FFFFFFFF FFFFFFFF | 00000000 00000000 0000 | A112FFC7 2F68417B |
| FFFFFFFF FFFFFFFF | FFFFFFFF FFFFFFFF FFFF | 3333DCD3 213210D2 |

## Appendix II

Here we describe a key schedule for a version of PRESENT that takes 128-bit keys. The user-supplied key is stored in a key register $K$ and represented as $k_{127}k_{126}\dots k_0$. At round $i$ the 64-bit round key $K_i = \kappa_{63}\kappa_{62}\dots\kappa_0$ consists of the 64 leftmost bits of the current contents of register $K$. Thus at round $i$ we have that:

$$K_i = \kappa_{63}\kappa_{62}\dots\kappa_0 = k_{127}k_{126}\dots k_{64}.$$

After extracting the round key $K_i$, the key register $K = k_{127}k_{126}\dots k_0$ is updated as follows.

1. $[\mathtt{k}_{127}\mathtt{k}_{126}\dots\mathtt{k}_1\mathtt{k}_0] = [\mathtt{k}_{66}\mathtt{k}_{65}\dots\mathtt{k}_{68}\mathtt{k}_{67}]$
2. $[\mathtt{k}_{127}\mathtt{k}_{126}\mathtt{k}_{125}\mathtt{k}_{124}] = S[\mathtt{k}_{127}\mathtt{k}_{126}\mathtt{k}_{125}\mathtt{k}_{124}]$
3. $[\mathtt{k}_{123}\mathtt{k}_{122}\mathtt{k}_{121}\mathtt{k}_{120}] = S[\mathtt{k}_{123}\mathtt{k}_{122}\mathtt{k}_{121}\mathtt{k}_{120}]$
4. $[\mathtt{k}_{66}\mathtt{k}_{65}\mathtt{k}_{64}\mathtt{k}_{63}\mathtt{k}_{62}] = [\mathtt{k}_{66}\mathtt{k}_{65}\mathtt{k}_{64}\mathtt{k}_{63}\mathtt{k}_{62}] \oplus \mathtt{round\_counter}$

Thus, the key register is rotated by 61 bit positions to the left, the left-most eight bits are passed through two PRESENT S-boxes, and the `round_counter` value $i$ is exclusive-ored with bits $\mathtt{k}_{66}\mathtt{k}_{65}\mathtt{k}_{64}\mathtt{k}_{63}\mathtt{k}_{62}$ of $K$ with the least significant bit of `round_counter` on the right.

## Appendix III

**Theorem 1.** Any 5-round differential characteristic of PRESENT has a minimum of 10 active S-boxes.

*Proof.* Recalling that the rounds are indexed from 1 to 31, consider five consecutive rounds of PRESENT ranging from $i-2$ to $i+2$ for $i \in [3\dots29]$. Let $D_j$ be the number of active S-boxes in round $j$. If $D_j \geq 2$, for $i-2 \leq j \leq i+2$, then the theorem trivially holds. So let us suppose that one of the $D_j$ is equal to one. We can distinguish several cases:

Case $D_i = 1$. The S-box of PRESENT is such that a difference in a single input bit causes a difference in at least two output bits (*cf.* the second design criterion). Thus $D_{i-1} + D_{i+1} \geq 3$. Using observation 1 above, all active S-boxes of round $i-1$ belong to the same group, and each of these active S-boxes have only a single bit difference in their output. So according to observation 2 we have that $D_{i-2} \geq 2D_{i-1}$. Conversely, according to observation 3, all active S-boxes in round $i+1$ belong to distinct groups and have only a single bit difference in their input. So according to observation 4 we have that $D_{i+2} \geq 2D_{i+1}$. Together this gives $\sum_{j=i-2}^{i+2} D_j \geq 1 + 3 + 2 \times 3 = 10$.

Case $D_{i-1} = 1$. If $D_i = 1$ we can refer to the first case, so let us suppose that $D_i \geq 2$. According to observation 3 above, all active S-boxes of round $i$ belong to distinct groups and have only a single bit difference in their input. Thus, according to observation 4, $D_{i+1} \geq 2D_i \geq 4$. Further, all active S-boxes in round $i+1$ have only a single bit difference in their input and they are distributed so that at least two groups of S-boxes contain at least one active S-box. This means that $D_{i+2} \geq 4$ and we can conclude that $\sum_{j=i-2}^{i+2} D_j \geq 1 + 1 + 2 + 4 + 4 = 12$.

Case $D_{i+1} = 1$. If $D_i = 1$ we can refer to the first case. So let us suppose that $D_i \geq 2$. According to observation 1 above, all active S-boxes of round $i$ belong to the same group and each of these active S-boxes has only a single bit difference in their output. Thus, according to observation 2, $D_{i-1} \geq 2D_i \geq 4$. Further, all active S-boxes of round $i-1$ have only a single bit difference in their output, and they are distributed so that at least two groups contain at least two active S-boxes. Thus, we have that $D_{i-2} \geq 4$ and therefore that $\sum_{j=i-2}^{i+2} D_j \geq 4 + 4 + 2 + 1 + 1 = 12$.

Cases $D_{i+2} = 1$ or $D_{i-2} = 1$. The reasoning for these cases is similar to those for the second and third cases.

The theorem follows. □

## Appendix IV

**Theorem 2.** Let $\epsilon_{4\mathrm{R}}$ be the maximal bias of a linear approximation of four rounds of PRESENT. Then $\epsilon_{4\mathrm{R}} \leq \frac{1}{2^7}$.

*Proof.* Recall that Matsui's piling-up lemma [32] estimates the bias of a linear approximation involving $n$ S-boxes to be

$$2^{n-1} \prod_{i=1}^{n} \epsilon_i,$$

where the values $\epsilon_i$ are the individual bias of each (independent) S-box. According to the design principles of PRESENT, the bias of all linear approximations is less than $2^{-2}$ while the bias of any single-bit approximation is less than $2^{-3}$. Let $\epsilon_{4\mathrm{R}}^{(j)}$ denote the bias of a linear approximation over 4 rounds involving $j$ active S-boxes. Now consider the following three cases.

1. Suppose that each round of a four-round linear approximation has exactly one active S-box. Then the bias of each of the two S-boxes in the middle

rounds is at most 1/8 and the overall bias for a four round approximation can be bounded as follows:

$$\epsilon_{4\mathrm{R}}^{(4)} \leq 2^3 \times (2^{-3})^2 \times (2^{-2})^2 = 2^{-7}.$$

2. Suppose, instead, that there are exactly five active S-boxes over four rounds. Then by the grouping of S-boxes in Figure 3, the active S-boxes over three consecutive rounds cannot form the pattern 1-2-1. For this to happen, the two active S-boxes in the middle round are activated by the same S-box and must therefore belong to two different groups of S-boxes. But if this is the case they couldn't activate only one S-box in the following round. Consequently the number of active S-boxes is either 2-1-1-1 or 1-1-1-2, so that
$$\epsilon_{4\mathrm{R}}^{(5)} \leq 2^4 \times (2^{-3}) \times (2^{-2})^4 = 2^{-7}.$$

3. Finally, suppose that there are more than five active S-boxes. Thus

$$\epsilon_{4\mathrm{R}}^{(j)} \leq 2^{j-1} \times (2^{-2})^j = 2^{-j-1} \leq 2^{-7} \text{ for } j > 5.$$

   The equality is theoretically attainable for $j = 6$. This is a strict inequality for all other $j$'s.

The theorem follows. $\square$