A Brief Comparison of Simon and Simeck

Stefan Kölbl, Arnab Roy
{stek,arroy}@dtu.dk

DTU Compute, Technical University of Denmark, Denmark

Abstract. SIMECK is a new lightweight block cipher design based on combining the SIMON and SPECK block cipher. While the design allows a smaller and more efficient hardware implementation, its security margins are not well understood. The lack of design rationals of its predecessors further leaves some uncertainty on the security of SIMECK.
In this work we give a short analysis of the impact of the design changes by comparing the lower bounds for differential and linear characteristics with SIMON. We also give a comparison of the effort of finding those bounds, which surprisingly is significant less for SIMECK while covering a larger number of rounds.
Furthermore, we provide new differentials for SIMECK which can cover more rounds compared to previous results on SIMON. Based on this we mount key recovery attacks on 19/26/33 rounds of SIMECK32/48/64, which also give insights on the reduced key guessing effort due to the different set of rotation constants.

Keywords: SIMON, SIMECK, differential cryptanalysis, block cipher

1 Introduction

SIMECK is a family of lightweight block ciphers proposed in CHES'15 by Yang, Zhu, Suder, Aagaard and Gongbased [7]. The design combines the SIMON and SPECK block ciphers proposed in [1], which leads to a more compact and efficient implementation in hardware. An alteration made by designers of SIMECK is the use of a different set of rotation constants. The designers of SIMON and SPECK do not provide rationales for the original choices apart from implementation aspects. These modifications are likely to have an impact on the security margins, which are often smaller for lightweight designs which can be delicate issue.

In this paper we give a first analysis on the impact of these design changes by comparing the bounds for differential and linear characteristics with the corresponding variants of SIMON. An unexpected advantage for SIMECK is, that it takes significant less time to find those while also covering more rounds (see Table 1).

Furthermore, we provide new differentials which cover 4 resp. 5 rounds for SIMECK48 and SIMECK64 with a slightly higher probability compared to previous results on SIMON. This is followed by key-recovery attacks for reduced round versions of SIMECK, covering more rounds (see Table 4).

Table 1. A comparison between the number of rounds for which lower bounds on the probability of differential and linear characteristics exist, the probability of differentials utilized in attacks and the best differential attacks on Simon and Simeck. Results contributed by this work are marked in bold.

Cipher	Rounds	Lower Bounds		Differentials		Key Recovery
		differential	linear	rounds	prob.	
Simon32/64	32	**32**	**32**	13	$2^{-28.79}$ [2]	21 [6]
Simeck32/64	32	**32**	**32**	**13**	$\mathbf{2^{-27.28}}$	**19**[1]
Simon48/96	36	**19**	**20**	16	$2^{-44.65}$ [5]	24 [6]
Simeck48/96	36	**36**	**36**	**20**	$\mathbf{2^{-43.65}}$	**26**
Simon64/128	44	15 [4]	**17**	21	$2^{-60.21}$ [5]	29 [6]
Simeck64/128	44	**40**	**41**	**26**	$\mathbf{2^{-60.02}}$	**33**

2 The Simeck Block Cipher

Simeck is a family of block ciphers with n-bit word size, where $n = 16, 24, 32$. Each variant has a block size of $2n$ and key size of $4n$ giving the three variants of Simeck: Simeck32/64, Simeck48/96 and Simeck64/128. As for each block size there is only one key size we will omit the key size usually.

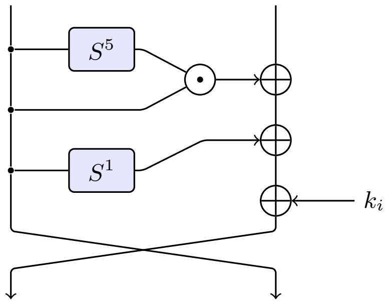


Fig. 1. The round function of Simeck.

The block cipher is based on the *Feistel* construction and the round function f is the same as in Simon apart from using $(5, 0, 1)$ for the rotation constants (as depicted in Figure 1). The key-schedule on the other hand is similar to Speck, reusing the round function to update the keys. The key K is split into four words

[1] The dynamic key guessing approach allows to cover two additional rounds at the beginning for Simon32.

(t_2, t_1, t_0, k_0) and the round keys $k_0, \dots, k_{r-1}$ are given by:

$$\begin{aligned} k_{i+1} &= t_i \\ t_{i+3} &= k_i \oplus f(t_i) \oplus C \end{aligned} \tag{1}$$

3 Properties of the Round Function

In [4] the differential and linear properties of Simon were studied, including variants using a different set of rotation constants. Following up on this work, we can use the same methods to analyze the round function of Simeck. This allows us to find lower bounds for the probability of a differential characteristic resp. square correlation of a linear characteristic for a given number of rounds.

We carried out experiments for the parameter set of Simeck using CryptoSMT[2] to find the optimal differential and linear characteristics for Simeck32, Simeck48 and Simeck64 and compare it with the results on Simon. The results of this experiment are given in Figure 2. The bounds on the square correlation for linear characteristics are given in the Appendix.

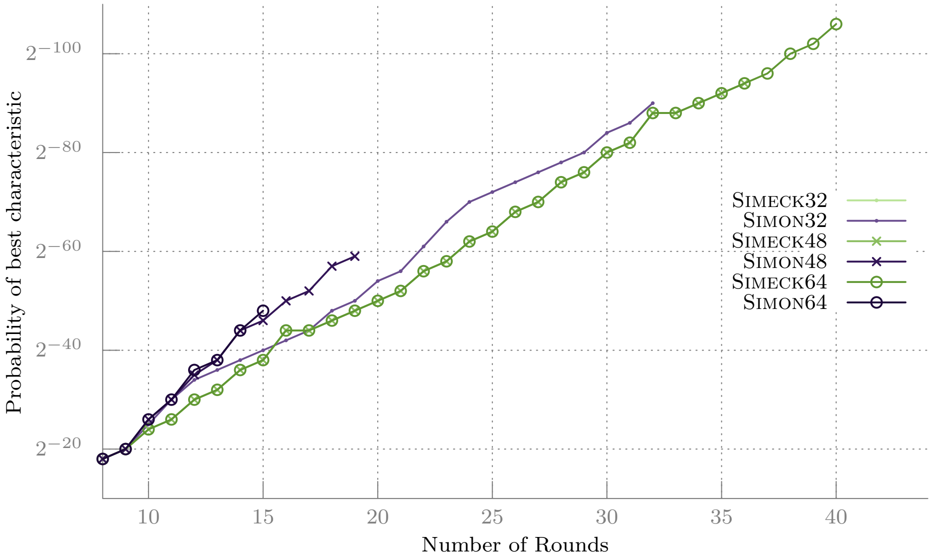


Fig. 2. Lower bounds on the probability of the best differential characteristics for variants of Simon and Simeck. For the different variants of Simeck the bounds are the same.

[2] CryptoSMT https://github.com/kste/cryptosmt Version: 70794d83

In our experiments we noticed that the different set of rotation constants plays a huge role in the running time of the SMT solver. For instance finding the bounds in Figure 2 took 51 hours for Simon32 and 10 hours for Simeck32[3]. Especially for larger block sizes it allows us to provide bounds for a significant larger number of rounds including full Simeck48. For Simon64 computing the bounds up to 15 rounds takes around 19 hours, while the same process only takes around 30 minutes for Simeck64. We computed the bounds for Simeck64 up to round 40 in around 53 hours.

4 Differentials for SIMECK

As noted in previous works Simon shows a strong differential resp. linear hull effect, which invalidates an often made assumption that the probability of the best characteristic can be used to estimate the probability of the best differential. Therefore bounds on differential and linear characteristics have to be treated with caution. The choice of constants for Simon-like round functions also plays a role in this as shown in [4].

For obtaining good differentials first we find the best characteristic for a given number of rounds of Simeck using CryptoSMT [3] and then find a large set of characteristics with the same input/output difference. The results of these experiments are summarized in Table 2. The differential for Simeck32 is not based on the best characteristic, as it would have drawbacks in the key recovery attack. However, the probability of this differential is even higher compared to differentials based on the best characteristic, as there are 8 characteristic with a probability of 2^{-36} contributing to the differential.

If we compare those with previous results on Simon we can cover more rounds. The best previous differential attack by Wang, Wang, Jia and Zhao [6] utilizes a 13-round differential for Simon32, a 16-round differential for Simon48 and a 21-round differential for Simon64. We show that almost with the same or slightly better probability (Table 1) differentials can be found for a higher number of rounds for both Simeck48 and Simeck64.

While we let our experiments run for a few days, the probability only improves marginal after a short time. For instance, for Simeck32 and Simeck48 the estimates after three minutes are only 2^{-2} lower than the final results and after two hours the improvements are very small. Some additional details on the differential utilized in the key-recovery attack on Simeck48 can be found in the Appendix 5, including the exact running times.

4.1 Choosing a good differential for attacks

For an attack we want a differential with a high probability, but also the form of the input/output difference can have an influence on the resulting attack complexity. Ideally we want differentials with a spare input/output difference

[3] Using Boolector 2.0.1. running on an Intel Xeon X5650 2.66GHz 48GB RAM (1 core).

Table 2. Overview of the differentials we found for Simeck which can likely be used to mount attacks. The probability is given by summing up all characteristics up to probability $2^{\max}$ taking a time t.

Cipher	Rounds	Δ_{in}	Δ_{out}	$\log_2(p)$	max	t
Simeck32	13	$(8000, 4011)$	$(4000, 0)$	-27.28	-49	17h
Simeck48	20	$(20000, 450000)$	$(30000, 10000)$	-43.65	-98	135h
Simeck48	20	$(400000, e00000)$	$(400000, 200000)$	-43.65	-74	93h
Simeck48	21	$(20000, 470000)$	$(50000, 20000)$	-45.65	-100	130h
Simeck64	25	$(2, 40000007)$	$(40000045, 2)$	-56.78	-90	110h
Simeck64	26	$(0, 4400000)$	$(8800000, 400000)$	-60.02	-121	120h

resp. of the form $(x, 0) \rightarrow (0, x)$. When expanding such a differential it leads to a truncated differential with fewer unknown bits. However, for 20-round Simeck48 the best characteristics with this pattern only has a probability of 2^{-62} and for $(x, x) \rightarrow (0, x)$ it is 2^{-54}. The corresponding differentials are not usable for an attack in this case. Therefore, we do not use these restrictions and use the 20-round characteristics with highest probability. For Simeck48 there are 768 such characteristics with a probability of 2^{-50} (32 rotation invariant) and we choose the one where the input/output difference is most sparse. The corresponding truncated differential obtained by extending in both rounds is given in Table 3.

Table 3. Truncated differential obtained by extending $(400000, e00000) \xrightarrow{20} (400000, 200000)$ in both directions until all bits are unknown.

Round	ΔL	ΔR	$*$	$*$
-5	`***0***0****************`	`************************`	22	24
-4	`***000000***0***********`	`***0***0****************`	17	22
-3	`***00000000000***0****1*`	`***000000***0***********`	11	17
-2	`***0000000000000000***01`	`***00000000000***0****1*`	6	11
-1	`111000000000000000000000`	`***0000000000000000***01`	0	6
0	`010000000000000000000000`	`111000000000000000000000`	0	0
20	`010000000000000000000000`	`001000000000000000000000`	0	0
21	`1*100000000000000000*000`	`010000000000000000000000`	2	0
22	`***000000000000*000***01`	`1*100000000000000000*000`	7	2
23	`***0000000*000***0****1*`	`***000000000000*000***01`	12	7
24	`***00*000***0***********`	`***0000000*000***0****1*`	18	12
25	`***0***0****************`	`***00*000***0***********`	22	18
26	`************************`	`***0***0****************`	24	22

5 Recovering the Key

In this section we describe the key recovery attack on SIMECK48 based on the differential given in Table 2. The input difference to round r is denoted as Δ^r and k_r denotes the round key for round r. The difference in the left resp. right part of the state we denote as ΔL^r and ΔR^r.

5.1 Attack on 26-round Simeck48

Our attack on 26-round SIMECK48 uses four 20-round differentials in a similar way as in [2]. Let D_i denote the differentials

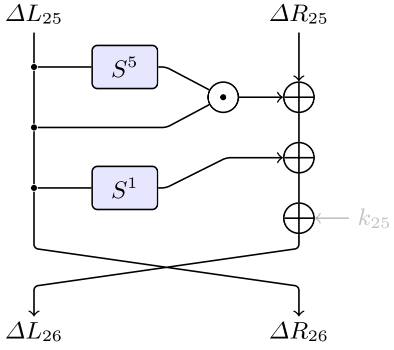
$$\begin{aligned}
\mathsf{D}_1 &: (\texttt{400000, e00000}) \to (\texttt{400000, 200000})\\
\mathsf{D}_2 &: (\texttt{800000, c00001}) \to (\texttt{800000, 400000})\\
\mathsf{D}_3 &: (\texttt{000004, 00000e}) \to (\texttt{000004, 000002})\\
\mathsf{D}_4 &: (\texttt{000008, 00001c}) \to (\texttt{000008, 000004})
\end{aligned}$$

each having probability $\approx 2^{-44}$. We add 4 rounds at the end by looking at the truncated differential and 2 rounds on top (see Table 3). The truncated difference at round 0 for each differential is given by

```
***00000000000000000***01, ***00000000000***0****1*
**000000000000000000***01*, **000000000000***0****1**
000000000000000***01***0, 0000000000***0****1****0
00000000000000***01***00, 000000000***0****1****00  .
```

By identifying the unknown and known bit positions in these differentials we can construct a set of 2^{31} plaintext pairs where the bit positions corresponding to the aligned 0s in the truncated differentials are fixed to an arbitrary value for all plaintexts. By guessing 6 round key bits we can also identify the 2^{31} pairs satisfying the difference $(\Delta L^2, \Delta R^2)$ after the first two round encryption. Hence we can get 4 sets of 2^{31} pairs of plaintexts where the difference is satisfied after the first two rounds of encryption. By varying the fixed bit positions we can get 4 sets of 2^{46} pairs of plaintexts, each satisfying the difference after two rounds for each key guess.

Filtering the pairs First we encrypt the 2^{46} plaintext pairs. Then we unroll the last round and use the truncated differential to verify if the pair is still valid. This is possible due to the last key addition not having any influence on the difference $(\Delta L^{25}, \Delta R^{25})$. As there are $12 + 17$ bits known in this round we will have $2^{46-29} = 2^{17}$ plaintext pairs left.



ΔL_{25}	`***0000000*000***0****1*`	12
ΔR_{25}	`***000000000000*000***01`	17

For each pair $(\Delta L', \Delta R')$ obtained:

1. Compute $\Delta R'_{25} = f(\Delta R'_{26}) \oplus \Delta L_{26}$[4].
2. Check if $\Delta L'_{25} = \Delta L_{25}$ and $\Delta R'_{25} = \Delta R_{25}$.

Fig. 3. Filtering for the correct pairs which we use in the key guessing part.

Key guessing In the key guessing phase we guess the necessary round key bits (or linear combination of round key bits) to verify the difference at the beginning of round 22, i.e. Δ^{22}. For each differential we counted that a total of 25 round key bits and linear combinations of round key bits are necessary to be guessed during this process. We describe this process for one round in Figure 4.

An interesting difference to Simon in the key guessing part is that the required number of key guesses is much lower, as many bits required to guess coincide when partially recovering the state which can reduce the overall complexity. This is always the case if one of the rotation constants is zero, but similar effects can occur with other choices as well.

For the key guessing part, we keep an array of 2^{25} counters and increment a counter when it is correctly verified with the difference after partial decryption of the ciphertext pairs. For each differential we can verify the remaining $19(= 48-29)$ bits with the key guessing process. For the 2^{25} counters we expect to have $(2^{17} \times 2^{25})/2^{19} = 2^{23}$ increments. The probability of a counter being incremented is $2^{23}/2^{25} = 2^{-2}$. Since 4 correct pairs are expected to be among the filtered pairs, the expected number of counters having 4 increments is $(\frac{1}{4})^4 \times 2^{23} = 2^{15}$.

We observe that there are 14 common key guesses between the differentials D_1 and D_2. Hence combining the corresponding counters T_1 and T_2 we can get $2^{15} \times 2^{15}/2^{14} = 2^{14}$ candidates for 36 bits. Continuing in the same way with D_3 we get $2^{15} \times 2^{14}/2^{15} = 2^{14}$ candidates for 46 bits and finally with D_4 we obtain $2^{15} \times 2^{14}/2^{22} = 2^7$ candidates for 49 bits. For the remaining 47 bits we perform an exhaustive search. Hence the total number of key guesses are $2^{47+7} = 2^{54}$.

The complexity for the partial decryption is $2^{17} \times 2^{25} \times \frac{4}{26} \approx 2^{40}$. Hence the complexity of the key recovery attack is 2^{54}.

Since the key recovery is performed for each of the 4 differentials and for each 2^6 round key guesses the overall complexity of the attack is $2^{54} \times 2^8 = 2^{62}$.

[4] The key has no influence on the input to the non-linear function in the last round.

5.2 Key Recovery forSimeck32

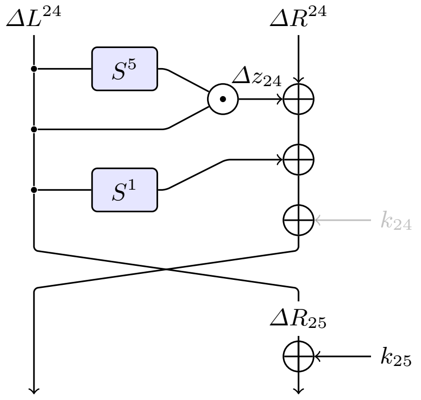
For Simeck32 we also use 4 differentials

$$\begin{aligned}
\mathsf{D}_1 &: \texttt{(8000, 4011)} \to \texttt{(4000, 0000)}\\
\mathsf{D}_2 &: \texttt{(0001, 8022)} \to \texttt{(8000, 0000)}\\
\mathsf{D}_3 &: \texttt{(0008, 0114)} \to \texttt{(0004, 0000)}\\
\mathsf{D}_4 &: \texttt{(0010, 0228)} \to \texttt{(0008, 0000)}
\end{aligned}$$

each having probability $\approx 2^{-28}$ (for the truncated differences see Table 6). We add two rounds at the top of the 13-round differential and identify a set of 2^{30} pairs of plaintexts each satisfying the specific difference $(\Delta L^2, \Delta R^2)$ after the first two round encryption. Identifying a set of plaintext pairs requires to guess 6 key bits.

Filtering We can filter some wrong pairs by unrolling the last round and verifying the truncated difference (with 18 known bits) at the beginning of the last round. This will leave us with $2^{30-18} = 2^{12}$ pairs.

Key guessing We counted that 18 round key bits and linear combination of round-key bits are necessary to guess for verifying the difference at the end of round 14. We use the same method as described for Simeck48 during this phase. Out of the filtered pairs we expect to get 4 correct pairs (those follow the 13-round differential). Hence the number of candidates for 18 key bits (and linear combinations) are 2^8. By combining all the four differentials we expect to get 1



ΔR_{25}	`***000000000000*000***01`
$S^5(\Delta R_{25})$	`0000000000*000***01***00`
Δz_{24}	`***0000000*000***0****0*`
ΔR_{24}	`1*100000000000000000*000`

Key filtering:

1. Find bits s.t. $\Delta z_{24} = *$ and $\Delta R_{24} \neq *$.
2. Guess corresponding bits in k_{25}.
3. Check $\Delta z_{24} = \Delta R_{24} \oplus S^1(R_{25}) \oplus \Delta L_{25}$.

Fig. 4. Outline of the process of key guessing and filtering for a single round.

key candidate for 37 bits. For the remaining 27 bits of the last four round keys we perform exhaustive search. Hence the complexity for the key guessing is 2^{27}.

The complexity of the partial decryption (for the last 4 rounds) is $2^{12} \times 2^{18} \times \frac{4}{19} \approx 2^{28}$ which is the dominating part of the complexity.

Since we perform the key recovery for each differential and for each 6-bit round key guesses of the first two rounds the overall complexity of the attack is $2^{28+8} = 2^{36}$.

5.3 Key Recovery for Simeck64

We use the following 4 differentials for SIMECK64

$$\begin{aligned}
\mathsf{D}_1 &: (\texttt{0, 04400000}) \rightarrow (\texttt{08800000, 00400000})\\
\mathsf{D}_2 &: (\texttt{0, 44000000}) \rightarrow (\texttt{88000000, 04000000})\\
\mathsf{D}_3 &: (\texttt{0, 40000004}) \rightarrow (\texttt{80000008, 40000000})\\
\mathsf{D}_4 &: (\texttt{0, 00000044}) \rightarrow (\texttt{00000088, 00000004})
\end{aligned}$$

each having probability $\approx 2^{-60}$ (for the truncated differences see Table 7). We add two rounds at the top of the 26 round differential and identify a set of 2^{62} pairs of plaintexts by guessing 4 round key bits from the first two rounds.

Filtering wrong pairs We add 5 round truncated difference at the end of the 26 round differential. The last round may be unrolled to verify the difference at the beginning of the last round. This helps to filter some wrong pairs using the known bits of the truncated difference and after filtering we are left with $2^{62-30} = 2^{32}$ pairs of plaintext out of which we expect 2^2 correct pairs (those followed 26 round differential).

Key guessing In this phase we guess the necessary key bits (or linear combination of key bits) from last four rounds to verify the difference at the beginning of round 28. We counted that 56 key bits are necessary to guess for verifying $(\mathit{\Delta} L^{28}, \mathit{\Delta} R^{28})$. Out of the filtered pairs we expect to get 4 correct pairs (those follow the 26-round differential). Hence the number of candidates for 56 key bits are 2^{36}. By combining all the four differentials we expect to get $2^{144-122} = 2^{22}$ key candidates for 102 bits. For the remaining 26 bits of the last four round keys we perform exhaustive search. Hence the complexity for the key guessing is $2^{26+22} = 2^{48}$.

The complexity of the partial decryption (for last 4 rounds) is $2^{32} \times 2^{56} \times \frac{5}{33} \approx 2^{86}$ which is the dominating part of the complexity.

Since we perform the key recovery for each differential and for each 6-bit round key guesses of the first two rounds the overall complexity of the attack is $2^{86+10} = 2^{96}$.

Table 4. Comparison of the attacks on Simeck.

Cipher	Rounds	Time	Data	Memory	Type
Simeck32/64	20/32	$2^{62.6}$	2^{32}	2^{56}	Imp. Differential [7]
Simeck32/64	19/32	2^{36}	2^{31}	2^{31}	Differential
Simeck48/96	24/32	$2^{94.7}$	2^{48}	2^{74}	Imp. Differential [7]
Simeck48/96	26/36	2^{62}	2^{47}	2^{47}	Differential
Simeck64/128	25/32	$2^{126.6}$	2^{64}	2^{79}	Imp. Differential [7]
Simeck64/128	33/44	2^{96}	2^{63}	2^{63}	Differential

6 Conclusion and Future Work

We gave a brief overview of the Simeck and Simon block cipher and their resistance against differential and linear cryptanalysis. From our comparison we can see that statistical attacks can cover a significant larger number of rounds for Simeck48 and Simeck64.

This also shows that the impact of small design changes in Simon-like block ciphers can be hard to estimate and requires a dedicated analysis, as the underlying design strategy is still not well understood. Especially for variants with a larger block size it is difficult to find lower bounds or estimate the effect of differentials.

In this sense Simeck has an unexpected advantage over Simon and Speck, as the analysis is simpler and requires less time with our approach. This is a property that is especially important in the light of not having cryptanalytic design documentation, nor design rationales for the constants regarding security available by the designers of Simon and Speck.

Our key recovery attacks still have a significant margin compared to generic attacks (see Table 4) in regard to time complexity, therefore it is likely they can be improved. For instance, if the dynamic key guessing approach [6] can be applied to Simeck with the same efficiency, this would likely extend our attacks by two more rounds.

References

1. Beaulieu, R., Shors, D., Smith, J., Treatman-Clark, S., Weeks, B., Wingers, L.: The SIMON and SPECK families of lightweight block ciphers. Cryptology ePrint Archive, Report 2013/404 (2013), http://eprint.iacr.org/
2. Biryukov, A., Roy, A., Velichkov, V.: Differential analysis of block ciphers SIMON and SPECK. In: Cid, C., Rechberger, C. (eds.) Fast Software Encryption, FSE 2014. Lecture Notes in Computer Science, vol. 8540, pp. 546–570. Springer (2015)
3. Kölbl, S.: CryptoSMT: An easy to use tool for cryptanalysis of symmetric primitives (2015), https://github.com/kste/cryptosmt

4. Kölbl, S., Leander, G., Tiessen, T.: Observations on the SIMON block cipher family. In: Advances in Cryptology - CRYPTO 2015. Lecture Notes in Computer Science, Springer, to appear.
5. Sun, S., Hu, L., Wang, P., Qiao, K., Ma, X., Song, L.: Automatic security evaluation and (related-key) differential characteristic search: Application to SIMON, PRESENT, LBlock, DES(L) and other bit-oriented block ciphers. In: Sarkar, P., Iwata, T. (eds.) Advances in Cryptology - ASIACRYPT 2014. Lecture Notes in Computer Science, vol. 8873, pp. 158–178. Springer (2014)
6. Wang, N., Wang, X., Jia, K., Zhao, J.: Differential attacks on reduced simon versions with dynamic key-guessing techniques. Cryptology ePrint Archive, Report 2014/448 (2014), http://eprint.iacr.org/
7. Yang, G., Zhu, B., Suder, V., Aagaard, M.D., Gong, G.: The simeck family of lightweight block ciphers. In: Cryptographic Hardware and Embedded Systems - CHES 2015. Springer (2015), to appear.

A Bounds for Linear Characteristics

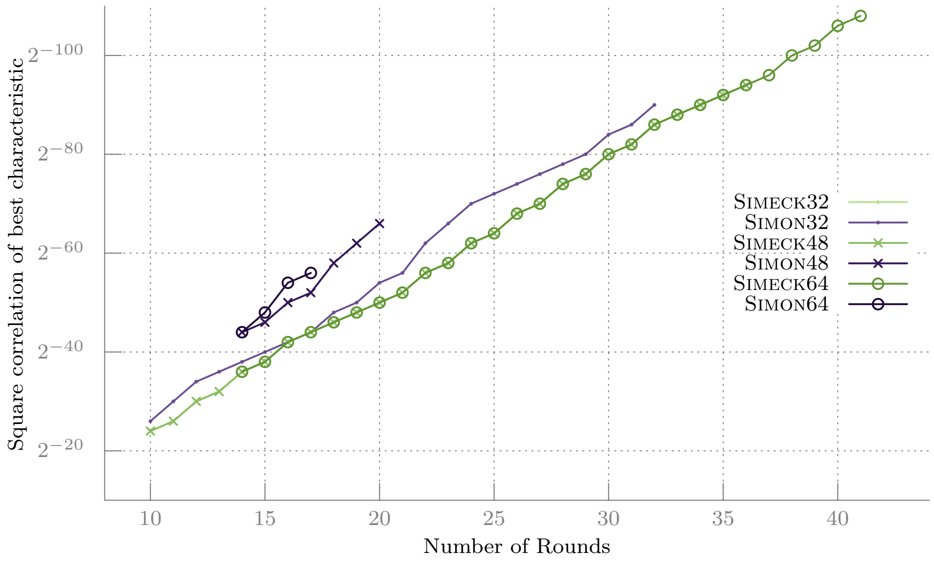


Fig. 5. Bounds for the best linear characteristics for variants of Simon and Simeck. For the different variants of Simeck the bounds are the same.

Table 5. Number of characteristics and time to find them for the SIMECK48 differential $(400000, e00000) \xrightarrow{f^{20}} (400000, 200000)$.

Pr(Char)	#Char.	Pr(Differential)	t
−50	1	−50.0	3.72s
−51	0	−50.0	6.9s
−52	12	−48.0	19.78s
−53	6	−47.7520724866	31.77s
−54	80	−46.7145977811	42.62s
−55	68	−46.4301443917	55.68s
−56	413	−45.804012702	77.58s
−57	484	−45.5334136623	104.69s
−58	1791	−45.1367816524	180.02s
−59	2702	−44.8963843436	265.5s
−60	7225	−44.6271009401	528.39s
−61	12496	−44.4289288164	1068.95s
−62	28597	−44.2312406041	2603.59s
−63	52104	−44.0720542548	6146.77s
−64	111379	−43.9193398907	19276.9s
−65	207544	−43.7902765446	41938.08s
−66	238939	−43.7209043818	70720.98s
−67	228530	−43.6888725691	96657.81s
−68	229018	−43.6730860168	123706.38s
−69	276314	−43.6636455186	160688.8s
−70	271192	−43.6590352669	197354.41s
−71	269239	−43.6567522016	232641.34s
−72	267563	−43.6556191172	271083.28s
−73	266716	−43.6550547005	308072.68s
−74	227971	−43.6548135551	336027.17s

Table 6. Truncated differential for SIMECK32 obtained by extending $(8000, 4011) \xrightarrow{f^{13}} (4000, 0)$ in both directions until all bits are unknown.

Round	ΔL	ΔR	$*$	$*$
−4	`***0************`	`****************`	15	16
−3	`**000***0****1**`	`***0************`	11	15
−2	`0*0000*000***01*`	`**000***0****1**`	6	11
−1	`0100000000010001`	`0*0000*000***01*`	0	6
0	`1000000000000000`	`0100000000010001`	0	0
13	`0100000000000000`	`0000000000000000`	0	0
14	`1*0000000000*000`	`0100000000000000`	2	0
15	`**00000*000**001`	`1*0000000000*000`	5	2
16	`***000**00***01*`	`**00000*000**001`	9	5
17	`***00***0*******`	`***000**00***01*`	13	9
18	`***0************`	`***00***0*******`	15	13
19	`****************`	`***0************`	16	15

Table 7. Truncated differential for SIMECK64 obtained by extending $(0, 4400000) \xrightarrow{f^{26}} (8800000, 400000)$ in both directions until all bits are unknown.

Round	ΔL	ΔR	$*$	$*$
−7	**0*****************************	********************************	31	32
−6	**00***0************************	**0*****************************	29	31
−5	**0000000***0*******************	**00***0************************	24	29
−4	**000000000000***0**************	**0000000***0*******************	19	24
−3	**00000000000000000***0*********	**000000000000***0**************	14	19
−2	1*0000000000000000000000***0****	**00000000000000000***0*********	8	14
−1	01000000000000000000000000000111	1*0000000000000000000000***0****	0	8
0	00000000000000000000000000000010	01000000000000000000000000000111	0	0
26	00001000100000000000000000000000	00000000010000000000000000000000	0	0
27	000**001*1000000000000000000000*	00001000100000000000000000000000	4	0
28	00***01***0000000000000000*000**	000**001*1000000000000000000000*	9	4
29	0****1****00000000000*000**00***	00***01***0000000000000000*000**	14	9
30	**********000000*000**00***0****	0****1****00000000000*000**00***	20	14
31	**********0*000**00***0*********	**********000000*000**00***0****	25	20
32	************00***0**************	**********0*000**00***0*********	29	25
33	************0*******************	************00***0**************	31	29
34	********************************	************0*******************	32	31