Differential Cryptanalysis of SPEEDY: Revised Security Evaluation and Improved Key Recovery Attack

Xinwei Yu[1,2], Wentao Zhang[1,2], Zairui Zhang[1,2] and Chunning Zhou[3]

[1] State Key Laboratory of Cyberspace Security Defense, Institute of Information Engineering, Chinese Academy of Sciences, Beijing, China
[2] School of Cyber Security, University of Chinese Academy of Sciences, Beijing, China
[3] Nanyang Technological University, Singapore

Abstract

The resistance of a block cipher against differential cryptanalysis (DC) is a fundamental criterion for evaluating its security. SPEEDY is a family of low-latency SPS-type block ciphers designed by Leander et al. The designers presented two evaluations of SPEEDY against DC: the first restricted 1-bit-to-1-bit differential propagations through the S-box and used the *higher-order* branch numbers to estimate the maximum expected differential probability (EDP) of differential trails; the second imposed a structural restriction and assumed that all 1-bit-to-1-bit propagations through an active S-box are possible to estimate the minimum number of active S-boxes. Zhang later exhibited a counterexample to the designers' first evaluation and introduced the notion of *partition branch number*, relaxing some of the propagation restrictions to allow multi-bit-to-1-bit transitions on some S-box layers and refining the evaluation. However, these evaluations consider only restricted spaces of differential characteristics and therefore inaccurately estimate the security of SPEEDY against DC, which means that the resistance of SPEEDY against DC remains unclear so far. To address this problem, we propose an efficient search method based on *search patterns*. Unlike previous evaluations, our method imposes no a priori structural restrictions on the allowed differential characteristics; it uses computational pruning to keep the search tractable and performs a conservative security evaluation of the pruned region, thereby providing an accurate evaluation over the global space. At the framework level, we decompose an r-round differential characteristic into an $(r-1)$-round *core trail* and an r-round *extended trail*. By extending a core trail independently at both ends, the two additional S-box layers are handled by two separate models, each containing only one S-box layer and solvable efficiently. At the implementation level, we further introduce a *short-round reuse via hash-map concatenation* technique, which concatenates short-round core trails to construct longer-round core trails, thereby avoiding expensive long-round MILP models. For 4 rounds, we narrow the minimum number of active S-boxes from the previously reported 35 to [33, 34] and find a characteristic with 34 active S-boxes outside the previous search spaces. We also obtain the highest-probability differential characteristics known to date for 2 to 5 rounds. In particular, the 2-, 3-, and 4-round characteristics disprove the designers' branch-number-based security claims. Furthermore, using the quasidifferential framework, we evaluate the found characteristics and obtain several 5-round distinguishers outperforming those used in existing key recovery attacks in terms of EDP. Finally, we optimize the probabilistic extension parameters used in a

Emails: yuxinwei@iie.ac.cn (Xinwei Yu), zhangwentao@iie.ac.cn (Wentao Zhang), zhangzairui@iie.ac.cn (Zairui Zhang), chunning.zhou@ntu.edu.sg (Chunning Zhou)

This work is licensed under a "CC BY 4.0" license.
Date of this document: 2026-11-03.

previous key recovery attack, which yields the best known key recovery attack on SPEEDY-7-192, with data complexity $2^{173.48}$ and time complexity $2^{173.69}$ encryptions.

Keywords: SPEEDY, Differential cryptanalysis, Security evaluation, Key recovery

1 Introduction

The SPEEDY family of block ciphers [LMMR21], designed by Leander et al. in 2021, stands out for its low-latency in hardware. Its components were carefully chosen to minimize hardware delay, in particular its optimized 6-bit S-box, which enables high encryption speed while maintaining strong cryptographic properties. Unlike classical substitution-permutation (SP) constructions with one S-box layer per round, SPEEDY employs a substitution-permutation-substitution (SPS) structure, where each round contains two SubBox (SB) layers connected by a linear ShiftColumns (SC) layer.

Together with the specification and design rationale, the designers of SPEEDY provided two security evaluations against differential cryptanalysis. First, the designers treated the composition SB ∘ SC ∘ SB as a single transformation and considered only 1-bit-to-1-bit differential propagations through the S-boxes. They then used the (higher-order) branch number bn_r to have an overview about the minimum number of active instances of the composite transformation over r rounds. For SPEEDY-r-192, their final estimates are $\mathrm{bn}_2 = 8$, $\mathrm{bn}_3 = 13$, $\mathrm{bn}_4 = 20$, $\mathrm{bn}_5 = 25$, and $\mathrm{bn}_6 = 32$. Since the maximum differential probability of an active instance of the composite transformation is 2^{-6}, they estimated the upper bound on differential probability of an r-round characteristic by $2^{-6 \cdot \mathrm{bn}_r}$, yielding 2^{-48}, 2^{-78}, 2^{-120}, 2^{-150}, and 2^{-192} for $r = 2, 3, 4, 5, 6$, respectively. However, the 1-bit-to-1-bit restriction excludes multi-bit propagations through the S-boxes and therefore does not cover the unrestricted differential search space. Indeed, Zhang [Zha24] later exhibited a 2-round differential characteristic with only 13 active S-boxes by exploiting such multi-bit propagations, showing that the designers' branch-number-based estimate does not hold globally. Separately, the designers gave another estimate for the minimum number of active S-boxes under an assumption and a structural restriction: all 1-bit-to-1-bit differential propagations through an active S-box are possible, and every differential state is restricted to contain at most 8 active S-boxes. Under these conditions, they derived minimum active S-box counts of 13, 23, and 35 for 2-, 3-, and 4-round characteristics, respectively.

Zhang [Zha24] introduced the notion of partition branch number to relax the designers' 1-bit-to-1-bit restriction and obtain tighter estimates of the minimum number of active S-boxes for SPEEDY. However, the method still restricts 1-bit-to-1-bit differential propagation in some SB layers and therefore does not cover the global search space. For example, in the 2-round case, taking the MC layer of the first round as the center, in both the forward and backward directions, Zhang still restricts the first SB layer to 1-bit-to-1-bit differential propagations, while allowing multi-bit-to-1-bit differential propagations in the second SB layer. In addition, Zhang also assumed that all 1-bit-to-1-bit differential propagations through an active S-box are possible. Under this restricted model, Zhang computed the partition branch numbers for 2, 3, 4, 5, 6, and 7 rounds as 13, 23, 35, 45, 57, and 67, respectively. Since the maximum differential probability of an active S-box is 2^{-3}, these numbers imply larger upper bounds on differential probability than the designers' branch-number-based estimates, namely 2^{-39}, 2^{-69}, 2^{-105}, 2^{-135}, 2^{-171}, and 2^{-201} for 2 to 7 rounds, respectively. In addition, Zhang searched for concrete differential characteristics using the actual difference distribution table of the SPEEDY S-box. The best probabilities found by Zhang for 2-, 3-, 4-, 5-, 6-, and 7-round characteristics are $2^{-46.2}$, $2^{-76.72}$, $2^{-129.2}$, $2^{-170.0}$, $2^{-216.0}$, and $2^{-266.2}$, respectively. Among these, the 2- and 3-round results directly disprove the designers' branch-number-based security claims [LMMR21].

The first claimed full-round key recovery attack on SPEEDY-7-192 was proposed by Boura et al. in 2023 [BDBN23]. Subsequently, Beyne and Neyt showed in a note [BN24], using the

quasidifferential framework from [BR22], that the dominant differential characteristic used in the attack of [BDBN23] is invalid. Using the same framework, Boura et al. [BDG+25b] showed that the differential characteristic employed in the differential attack of [WNL+23] is also invalid; they further pointed out that the key recovery phase of the linear attack in [WNL+23] is incorrect. After these invalidations, Boura et al. [BDG+25b] and Beyne et al. [BN25] independently proposed valid key recovery attacks on SPEEDY-7-192, with the former achieving the lowest data and time complexities prior to our work.

Contributions. Previous evaluations of SPEEDY are subject to two opposite sources of inaccuracy: structural restrictions may exclude valid differential characteristics and thus overestimate the minimum number of active S-boxes, whereas assumptions on S-box transitions may admit impossible propagations and thus underestimate it. Consequently, the actual security of SPEEDY against differential cryptanalysis remains unclear. To address this gap, we develop an efficient search strategy that avoids a priori structural restrictions on differential characteristics and uses computational pruning together with conservative bounds on the pruned region to obtain an accurate estimates in the global space. We further explore how the resulting characteristics can be clustered into stronger differential distinguishers and leveraged to improve full-round key recovery attacks. Our main contributions are summarized as follows:

1. We propose a new search method for differential characteristics based on *search patterns* [OMA95, ZZC24]. Specifically, for r-round characteristics with total n active S-boxes, we enumerate all ordered compositions $(x_1, \ldots, x_{2r})$ of n, where x_i denotes the number of active S-boxes in the i-th SB layer. Each composition is a search pattern, which serves as the basic search unit. At the architectural level, we carefully decompose the SPEEDY differential characteristic into two subtrails, namely, a *core trail* corresponding to the pattern $(x_2, \ldots, x_{2r-1})$ and an *extended trail* corresponding to the pattern $(x_1, \ldots, x_{2r})$, which together form an efficient search framework. By extending $(r-1)$-round core trails at both ends to obtain r-round extended trails, our framework distributes the two additional SB layers into two separate models, each containing only one SB layer and solvable efficiently. At the implementation level, we further introduce a technique called *short-round reuse via hash-map concatenation* to search for core trails. Specifically, we build MILP models only for short rounds ($r = 1, 2$), and use hash maps to concatenate the resulting short-round solutions to construct long-round core trails ($r = 3, 4$). This avoids expensive MILP modeling for long rounds and reuses common short-round core patterns across multiple concatenations, thereby significantly reducing the overall search time.

2. Our search results disprove some of the security bounds given by the designers and provide a more accurate security evaluation of SPEEDY against differential cryptanalysis in the global space, in terms of both the minimum number of active S-boxes and the upper bounds on differential probability, as summarized in Table 1. For the minimum number of active S-boxes, we determine the exact minimum numbers for 2, 3, and 5 rounds as 13, 23, and 45, respectively. For 4 rounds, the minimum lies in the interval $[33, 34]$ (both previous evaluations report a minimum of 35), and we find a differential characteristic with 34 active S-boxes outside both previous evaluation frameworks; see Table 16 in Appendix C.1. For differential probability, we find the highest-probability differential characteristics known to date for 2, 3, 4, and 5 rounds, with probabilities $2^{-46.245}$, $2^{-76.15}$, $2^{-119.735}$, and $2^{-156.565}$, respectively. The 2-, 3-, and 4-round results disprove the the designers' branch-number-based security claims. Moreover, we determine upper bounds on the differential probability for 2, 3, 4, and 5 rounds as $2^{-46.245}$, 2^{-75}, 2^{-99}, and 2^{-135}, respectively. For 2

and 3 rounds, these upper bounds are obtained by exhaustive searches over all characteristics with 13, 14, and 15 active S-boxes, and with 23 and 24 active S-boxes, respectively; consequently, these bounds are tighter than those estimated solely from the minimum number of active S-boxes.

3. From an attack perspective, our search also identifies the best known differential characteristics and differentials in terms of the expected differential probability. We use the quasidifferential framework [BDG25a] to verify the validity of the differential characteristics and to compute their expected differential probabilities. As a result, we obtain the currently best known 2-, 3-, 4-, and 5-round differential characteristics, with expected differential probabilities of $2^{-46.245}$, $2^{-74.998}$, $2^{-117.32}$, and $2^{-152.565}$, respectively. Moreover, by clustering the 5-round characteristics according to their input and output differences, we obtain a number of 5-round distinguishers whose expected differential probabilities are higher than those of the 5-round distinguishers used in existing key recovery attacks [BDG+25b, BN25], including the best known 5-round distinguisher to date, with expected differential probability $2^{-151.766}$.

4. We obtain the currently best known key recovery attack on SPEEDY-7-192 in terms of both data and time complexity, with data complexity $2^{173.48}$ and time complexity $2^{173.69}$ encryptions. This improvement is achieved by optimizing the probabilistic extension parameters in the attack of [BDG+25b], which reduces $l_{\text{in}} + l_{\text{out}}$ from 32 to 31. This reduction lowers the probability cost of the extension and consequently reduces the data and time complexities of the data generation phase. Although this increases the number of message pairs entering the key recovery phase, the efficient key recovery procedure keeps the overall time complexity lower than that of the previous attacks.

Table 2 summarizes the attacks on SPEEDY-7-192 from both our work and all previously known works, and we mark the attacks that are incorrect. For differential attacks, invalidity comes from flaws in the distinguisher, while for the linear attack, the issue lies in the key recovery phase. Candidate 1 and Candidate 2 are the attacks proposed in this work.

Table 1: Comparison of security evaluations of SPEEDY against differential cryptanalysis in terms of minimum active S-box counts and differential probability bounds.

Rounds	Minimum active S-box count			Differential probability				
	Designers [LMMR21]	Zhang [Zha24]	This work	Designers [LMMR21]	Zhang [Zha24]		This work	
	$\#AS^*_{\min}$	$\#AS^*_{\min}$	$\#AS_{\min}$	P^*_{UB}	P_{BEST}	P^*_{UB}	P_{BEST}	P_{UB}
2	13	13	13	2^{-48}	$2^{-46.2}$	2^{-39}	$2^{-46.245}$	$2^{-46.245}$
3	23	23	23	2^{-78}	$2^{-76.72}$	2^{-69}	$2^{-76.15}$	2^{-75}
4	35	35	[33, 34]	2^{-120}	$2^{-129.2}$	2^{-105}	$2^{-119.735}$	2^{-99}
5	–	45	45	2^{-150}	2^{-170}	2^{-135}	$2^{-156.565}$	2^{-135}

$\#AS^*_{\min}$ and P^*_{UB} denote the results under restricted evaluations, whereas $\#AS_{\min}$ and P_{BEST} denote the global estimate. P_{BEST} denotes the best differential probability found in the corresponding search.

Organization. The remainder of this paper is organized as follows. Section 2 recalls the preliminaries, including the SPEEDY family and differential cryptanalysis. Section 3 presents our search method for differential characteristics and the revised security evaluation of SPEEDY against differential cryptanalysis, in terms of both the minimum number of active S-boxes and upper bounds on differential probability. Section 4 describes our full 7-round key recovery attack on SPEEDY-7-192. Section 5 concludes the paper.

Table 2: Summary of all known attacks on SPEEDY-7-192. KP = known-plaintext, CP = chosen-plaintext, CC = chosen-ciphertext.

Version	Rounds	Type	Data	Time	Memory	Flawed	Source
SPEEDY-7-192	7	Diff.	$2^{186.53}$ CC	$2^{187.39}$	2^{36}	✓	[WNL+23]
	7	Linear	$2^{188.5}$ KP	$2^{189.41}$	$2^{185.04}$	✓	[WNL+23]
	7	Diff.	$2^{187.28}$ CC	$2^{187.84}$	2^{42}	✓	[BDBN23]
	7	Diff.	2^{183} CC	2^{185}	2^{36}	–	[BN25]
	7	Diff.	$2^{174.53}$ CC	$2^{174.53}$	2^{66}	–	[BDG+25b]
	7	Diff.	$2^{173.48}$ CC	$2^{173.69}$	2^{72}	–	Candidate 1(Sect.4)
	7	Diff.	$2^{173.55}$ CC	$2^{173.76}$	$2^{66.06}$	–	Candidate 2(Sect.4)

2 Preliminaries

2.1 Specification of SPEEDY

SPEEDY is a family of ultra low-latency block ciphers proposed by Leander et al. [LMMR21]. An instance of the SPEEDY family is denoted as SPEEDY-r-$6l$, where r is the number of rounds and $6l$ is the block length as well as the key length. The parameters can be flexibly adjusted according to the requirements of encryption speed and security. The default values recommended by the designers are $r \in \{5, 6, 7\}$ and $l = 32$, i.e., a block length of 192 bits. The internal state x of SPEEDY is represented as a 32×6 two-dimensional rectangular array of bits, where the elements are denoted by $x[i][j]$ with i the row index and j the column index, $0 \le i < 32$, $0 \le j < 6$. The bit at position 0 is the most significant bit. Additionally, computations within a row and within a column are performed modulo 6 and modulo 32, respectively.

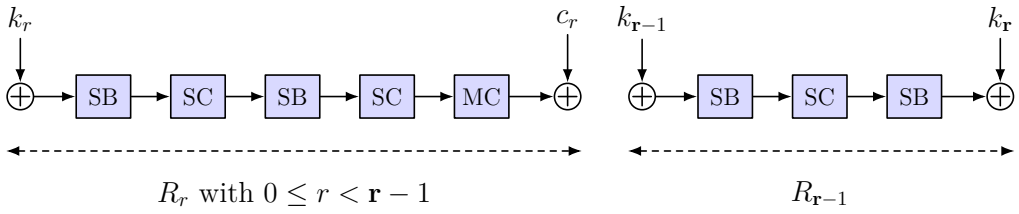

Figure 1: Illustration of the SPEEDY-r-192 round function.

Round function. The encryption structure of SPEEDY-r-192 is shown in Figure 1. The internal state is first initialized by filling it with the 192-bit plaintext (i.e., from left to right and top to bottom). Then, the internal state is encrypted for r rounds. The operations in the round function of the first $r - 1$ rounds are performed in the following order: AddRoundKey, SubBox, ShiftColumns, SubBox, ShiftColumns, MixColumns, AddRoundConstant. The round function of the r-th round is different: the last three operations (ShiftColumns, MixColumns, AddRoundConstant) are omitted and replaced by an AddRoundKey operation.

- **AddRoundKey (A_{k_r}):** The 192-bit round key k_r is XORed with the internal state:

$$y[i][j] = x[i][j] \oplus k_r[i][j], \quad \text{for } 0 \le i < 32,\ 0 \le j < 6.$$

- **SubBox (SB):** Apply a 6-bit S-box to each row of the state:

$$(y[i][0], \ldots, y[i][5]) = S(x[i][0], \ldots, x[i][5]), \quad \text{for } 0 \le i < 32.$$

The content of the S-box is presented in Table 13 of Appendix A.1.

- **ShiftColumns (SC):** Each column j of the state is rotated upward by j positions:

$$y[i][j] = x[i+j][j], \quad \text{for } 0 \le i < 32,\ 0 \le j < 6.$$

- **MixColumns (MC):** This operation is also applied column-wise. Specifically, each column of the output state is obtained by multiplying a binary matrix with the corresponding column of the input state. More precisely, each output bit is XORed from seven input bits:

$$\begin{aligned} y[i][j] =& x[i][j] \oplus x[i+\alpha_1][j] \oplus x[i+\alpha_2][j] \oplus x[i+\alpha_3][j] \oplus x[i+\alpha_4][j] \\ & \oplus x[i+\alpha_5][j] \oplus x[i+\alpha_6][j], \quad \text{for } 0 \le i < 32,\ 0 \le j < 6. \end{aligned}$$

  For $l = 32$, $\alpha = (1, 5, 9, 15, 21, 26)$.

- **AddRoundConstant (A_{c_r}):** The 192-bit round constant c_r is XORed with the internal state:

$$y[i][j] = x[i][j] \oplus c_r[i][j], \quad \text{for } 0 \le i < 32,\ 0 \le j < 6.$$

  The specific round constants can be found in the design paper [LMMR21].

Key schedule. The round key k_0 of the first round is initialized as the 192-bit master key. Then, the round key of each subsequent round is obtained from the previous round key by applying the bit-permutation PB, i.e., $k_r := PB(k_{r-1})$. The bit-permutation PB is presented in Table 14 of Appendix A.2.

2.2 Differential cryptanalysis

Differential cryptanalysis was introduced by Biham and Shamir in 1990 [BS90], which is one of the oldest and most powerful techniques for analyzing symmetric cryptographic primitives. Differential cryptanalysis mainly consists of two major steps: the first is to find a high-probability differential distinguisher, and the second is to perform a key recovery attack using that distinguisher.

2.2.1 Differential distinguisher

Definition 1 (Differential)**.** Let F be a function over $\mathbb{F}_2^n$. A *differential* of F is a pair of input and output differences $(\Delta_{\text{in}}, \Delta_{\text{out}}) \in \mathbb{F}_2^n \times \mathbb{F}_2^n$. The *differential probability* of $(\Delta_{\text{in}}, \Delta_{\text{out}})$ is defined as:

$$\mathrm{DP}_F(\Delta_{\text{in}}, \Delta_{\text{out}}) \coloneqq \frac{|\{X \in \mathbb{F}_2^n : F(X) \oplus F(X \oplus \Delta_{\text{in}}) = \Delta_{\text{out}}\}|}{2^n}.$$

Definition 2 (Differential Characteristic)**.** For an r-round iterated block cipher $E_k = F_{r-1} \circ \cdots \circ F_0$, an r-round *differential characteristic* is an ordered vector of intermediate differences $(\Delta_0, \Delta_1, \ldots, \Delta_r) \in (\mathbb{F}_2^n)^{r+1}$. Its differential probability is defined as

$$\mathrm{DPC}_{E_k}(\Delta_0, \Delta_1, \ldots, \Delta_r) := \Pr\left[\bigwedge_{i=0}^{r-1} F_i(x_i) \oplus F_i(x_i \oplus \Delta_i) = \Delta_{i+1}\right],$$

where x_0 uniform random on $\mathbb{F}_2^n$, and $x_i = F_{i-1}(x_{i-1})$ for $i = 1, \ldots, r$.

For an r-round iterated block cipher $E_k = F_{r-1} \circ \cdots \circ F_0$, the probability of a differential $(\Delta_{\text{in}}, \Delta_{\text{out}})$ can be obtained by summing the probabilities of all differential characteristics that share the same input difference Δ_{in} and output difference Δ_{out}. That is,

$$\text{DP}_{E_k}(\Delta_{\text{in}}, \Delta_{\text{out}}) = \sum_{\substack{\Delta_1,\ldots,\Delta_{r-1} \in \mathbb{F}_2^n \\ \Delta_0=\Delta_{\text{in}},\ \Delta_r=\Delta_{\text{out}}}} \text{DPC}_{E_k}(\Delta_0, \Delta_1, \ldots, \Delta_r).$$

Definition 3 (Expected Differential Probability)**.** For a keyed cipher E_k with key space $\mathbb{F}_2^k$, the differential probability of a characteristic may vary across different keys. The expected differential probability of a characteristic is the average of its probability over the entire key space, and is given by

$$\text{EDPC}_{E_k}(\Delta_0, \Delta_1, \ldots, \Delta_r) := \frac{1}{2^k} \sum_{k \in \mathbb{F}_2^k} \text{DPC}_{E_k}(\Delta_0, \Delta_1, \ldots, \Delta_r).$$

And the expected differential probability of a differential can be defined as:

$$\text{EDP}_{E_k}(\Delta_{\text{in}}, \Delta_{\text{out}}) = \sum_{\substack{\Delta_1,\ldots,\Delta_{r-1} \in \mathbb{F}_2^n \\ \Delta_0=\Delta_{\text{in}},\ \Delta_r=\Delta_{\text{out}}}} \text{EDPC}_{E_k}(\Delta_0, \Delta_1, \ldots, \Delta_r).$$

In differential cryptanalysis, a common assumption is the *hypothesis of stochastic equivalence* [LMM91]. This hypothesis states that for any fixed, randomly chosen key, the differential probability of a characteristic is close to its average over all keys. However, some studies [Knu92, AK18, Hey20] have shown that this assumption is not always valid in practice; that is, the differential probability of the same characteristic can vary significantly across different keys.

Quasidifferential framework. In 2022, Beyne and Rijmen introduced the quasidifferential framework [BR22], providing for the first time an accurate method for computing the differential probability of a characteristic under a fixed key. They defined the notion of quasidifferential trails; in essence, a quasidifferential trail characterizes a linear approximation of the valid plaintext pairs that follow the given differential characteristic. The differential probability of a characteristic is then computed via the correlation of the associated quasidifferential trails.

Definition 4 (Quasidifferential Trail)**.** Let E_k be a keyed iterated block cipher with $E_k = F_{r-1} \circ \cdots \circ F_0$, where each F_i is a round function. A *quasidifferential trail* of E_k is a sequence of $(r+1)$ mask-difference pairs $((u_0, \Delta_0), (u_1, \Delta_1), \ldots, (u_r, \Delta_r))$. The correlation of a quasidifferential trail is defined as

$$\text{Corr}_{E_k}\big((u_0, \Delta_0), (u_1, \Delta_1), \ldots, (u_r, \Delta_r)\big) = \prod_{i=0}^{r-1} \text{Corr}^{F_i}_{(u_i,\Delta_i),(u_{i+1},\Delta_{i+1})},$$

where $\text{Corr}^{F_i}_{(u_i,\Delta_i),(u_{i+1},\Delta_{i+1})}$ denotes the correlation of the mask-difference pair in the round function F_i, and is given by

$$\text{Corr}^{F_i}_{(u_i,\Delta_i),(u_{i+1},\Delta_{i+1})} = 2^{-n} \sum_{\substack{x \in \mathbb{F}_2^n \\ F_i(x) \oplus F_i(x \oplus \Delta_i) = \Delta_{i+1}}} (-1)^{u_i^\intercal x \,\oplus\, u_{i+1}^\intercal F_i(x)}.$$

Intuitively, this correlation quantifies the linear approximation of the valid message pairs that follow the differential transition $(\Delta_i \to \Delta_{i+1})$ through the round function F_i, with respect to the masks (u_i, u_{i+1}).

Theorem 1 ([BR22]). *Let E_k be a keyed iterated block cipher with round functions F_i for $i = 0, \ldots, r-1$. For a differential characteristic $(\Delta_0, \Delta_1, \ldots, \Delta_r)$, its differential probability is equal to the sum of the correlations of all quasidifferential trails that share the same intermediate differences $\Delta_0, \ldots, \Delta_r$. Specifically,*

$$\mathrm{DPC}_{E_k}(\Delta_0, \ldots, \Delta_r) = \sum_{u_1, \ldots, u_{r-1} \in \mathbb{F}_2^n} \mathrm{Corr}_{E_k}\big((u_0, \Delta_0), (u_1, \Delta_1), \ldots, (u_r, \Delta_r)\big),$$

where $u_0 = u_r = 0$.

In fact, the correlation of a quasidifferential trail can be expressed as the product of a key-dependent sign and an absolute correlation. Specifically,

$$\mathrm{Corr}_{E_k}\big((u_0, \Delta_0), (u_1, \Delta_1), \ldots, (u_r, \Delta_r)\big) = (-1)^{U^\intercal K} \cdot C,$$

where U is a selection mask vector derived from the masks at the key addition layers, K is a key-bit vector composed of the master key (or the round keys), and C is a key-independent quantity. Therefore, for a given key, the correlation of a quasidifferential trail can be completely determined; and then Theorem 1 can be used to obtain the exact differential probability of the differential characteristic under that fixed key.

In 2025, Boura, Derbez, and Germon extended the quasidifferential framework from fixed-key model to averaged-key model [BDG25a], providing an exact method for computing the expected differential probability of a characteristic. They treated the key as a variable equivalent to the message, incorporated the key schedule into the modeling, and directly integrated the key into the quasidifferential trails. Under this approach, the quasidifferential trails obtained by the model correspond exactly to the case $U \perp K$, i.e., the correlation is independent of the key. Therefore, summing the correlations of these quasidifferential trails yields the expected differential probability of the given characteristic.

In this paper, we use the term *differential probability* to refer to the probability computed by the traditional method, i.e., the product of the differential probabilities of all active S-boxes in the differential characteristic. The term *expected differential probability*, on the other hand, refers to the average probability of a differential characteristic over the entire key space, which is computed using the quasidifferential framework under the averaged-key model [BDG25a]. Additionally, the terms *maximum* and *minimum* refer to global extrema in the entire search space, whereas *best* refers to the optimal value found within our search (e.g., for the differential probability of r-round characteristics).

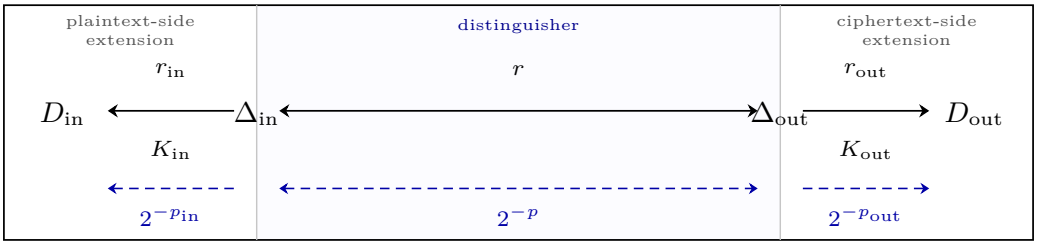

Figure 2: Key recovery in differential cryptanalysis.

2.2.2 Key recovery

After obtaining a high-probability differential distinguisher, the next step is to extend it on both sides to perform a key recovery attack, as illustrated in Figure 2. Let the input and output differences of the intermediate r-round distinguisher be Δ_{in} and Δ_{out}, respectively,

with differential probability 2^{-p}. On the plaintext side, the distinguisher input difference Δ_{in} can be extended backward by r_{in} rounds with probability $2^{-p_{\text{in}}}$, yielding a truncated difference in an affine subspace $D_{\text{in}} \subseteq \mathbb{F}_2^n$ of dimension d_{in}. Similarly, on the ciphertext side, we extend the distinguisher output difference Δ_{out} and define the corresponding parameters $r_{\text{out}}, p_{\text{out}}, D_{\text{out}}, d_{\text{out}}$. Additionally, we define $p' = p_{\text{in}} + p + p_{\text{out}}$. Typically, $p_{\text{in}} = p_{\text{out}} = 0$, meaning that the extensions on both sides are deterministic. However, in works such as [BDG+25b, SYC+24, AKM+24], probabilistic extensions are employed, i.e., $p_{\text{in}}, p_{\text{out}} \neq 0$. Although this incurs a certain probability cost, it can increase the numbers of inactive rows on the plaintext side and the ciphertext side after the extension, denoted by l_{in} and l_{out}, respectively, which significantly reduces the number of data pairs entering the key recovery phase, thereby lowering the overall complexity of the key recovery attack. Following [BDG+25b], we also adopt probabilistic extensions in our key recovery attack on SPEEDY-7-192. Without loss of generality, we describe the key recovery attack in the encryption direction.

Data generation phase. In differential attacks, attackers commonly use *structures* to obtain enough message pairs. A structure is a set of plaintexts that share fixed values in the inactive bits and take all possible values in the remaining d_{in} active bits. Such a structure contains $2^{d_{\text{in}}}$ plaintexts and can generate approximately $2^{2d_{\text{in}}-1}$ message pairs. On average, the number of message pairs required to obtain one correct pair satisfying the entire differential propagation path is about $2^{d_{\text{in}}+p'}$. In most cases, the number of message pairs generated by a single structure is insufficient to meet this requirement. Therefore, the number of structures needed is $2^s = 2^{p'+1-d_{\text{in}}}$. The data complexity of the data generation phase is $2^s \times 2^{d_{\text{in}}} = 2^{p'+1}$. When encrypting plaintexts using structures, the ciphertexts are stored in hash tables indexed by the values of the $n - d_{\text{out}}$ inactive bits, enabling pre-sieving of message pairs. And the memory complexity of the data generation phase is the size of one structure, i.e., $2^{d_{\text{in}}}$. Finally, about $2^{p'+d_{\text{in}}+d_{\text{out}}-n}$ message pairs $((P, C = E_k(P)), (P', C' = E_k(P')))$ satisfying $P \oplus P' \in D_{\text{in}}$ and $C \oplus C' \in D_{\text{out}}$ enter the subsequent key recovery phase.

Key recovery phase. Let K_{in} be the set of candidate key materials involved in the computation of Δ_{in} in the extended rounds on the plaintext side, and let K_{out} be the corresponding set on the ciphertext side, and define $K = K_{\text{in}} \cup K_{\text{out}}$. For a tuple $((P, C = E_k(P)), (P', C' = E_k(P')), k)$ with $k \in K$ satisfying $P \oplus P' \in D_{\text{in}}$ and $C \oplus C' \in D_{\text{out}}$, if under the action of k, partial encryption of (P, P') yields Δ_{in} and partial decryption of (C, C') yields Δ_{out}, then k is selected as a valid candidate for the key material. The key recovery process is essentially a matching procedure between data pairs $((P, C = E_k(P)), (P', C' = E_k(P')))$ and keys k. If the recovered key bits in K are insufficient to fully determine the master key, the remaining unknown key bits need to be exhausted by brute force. In practice, the above matching process can be viewed as a pipeline, where each step matches a subset k' of K with the current remaining tuples, enabling partial encryption (or decryption) that satisfies part of Δ_{in} (or Δ_{out}) under the action of k', until all key bits in K are recovered. It is worth noting that there is no universal method or formula for this step, as it is highly non-trivial. In our key recovery attack on SPEEDY-7-192, we adopt key recovery steps similar to that in [BDG+25b] which is generated by KyRyDi [BDD+24], an automated tool for computing efficient key recovery procedures for block ciphers whose linear layers consist of a bit permutation (for SPEEDY, the linear layers involved in the extended rounds on both sides of the distinguisher consist only of bit permutations).

3 Searching differential characteristics and re-evaluating the security of SPEEDY against differential cryptanalysis

The designers of SPEEDY [LMMR21] evaluated its security against differential cryptanalysis by deriving lower bounds on the number of active S-boxes through the (higher-order) branch number of the MC layer. They treat SB ∘ SC ∘ SB as a single transformation and ignore the intermediate SC layer, which can redistribute active bits between the two SB layers. Consequently, the resulting bounds may overestimate the security of SPEEDY. In this section, we introduce an efficient search method for differential characteristics and use it to re-evaluate the security of SPEEDY against differential cryptanalysis in terms of the minimum number of active S-boxes and upper bounds on the differential probability. Our experimental setup is summarized in Appendix B.

3.1 Overview of the search strategy

Searching for the differential characteristics of r-round SPEEDY by directly solving a single MILP model is prohibitively slow, as the model is large and grows rapidly with the number of rounds. We therefore adopt a *search pattern*-based approach [OMA95, ZZC24] and decompose each characteristic into core and extended trails, then search them by extending short trails into long ones, which keeps every MILP model small and the search efficient. These small models are solved with the Open Cryptanalysis Platform (OCP) [OCP], an open-source framework for the automated cryptanalysis of symmetric-key primitives. In addition, provably infeasible patterns are discarded beforehand, further reducing the number of patterns to be searched.

From security bounds to pattern search. To compute the minimum number of active S-boxes, we search for r-round characteristics with an increasing total number n of active S-boxes, starting from a small value, and return the first n that yields a characteristic. To find the best differential characteristic, we keep increasing n while tracking the current smallest weight $\overline{W}$ found so far. Since each active S-box has weight at least 3, a characteristic with n active S-boxes has weight at least $3n$. We therefore stop once $3n \geq \overline{W}$, at which point no unexamined characteristic can have a smaller weight. Both problems therefore reduce to searching for r-round differential characteristics with a prescribed *search pattern* [ZZC24], where each search pattern fixes the number of active S-boxes in every SB layer.

Decomposition of an r-round characteristic. As illustrated in Figure 3, an r-round differential characteristic is built from two nested sub-trails. We denote by st_j^i the j-th intermediate state of the i-th round. The $(r-1)$-round **core trail**, from st_3^0 of the first round to st_0^{r-1} of the last round (blue box), is the building block of our search. Adding one SB and one SC on each side extends it to an r-round **extended trail**, from st_1^0 of the first round to st_2^{r-1} of the last round (red box). Extending it further on both sides yields the r-round characteristic. Let x_i $(i = 1, \ldots, 2r)$ denote the number of active S-boxes in the i-th SB layer, so that $[x_1, \ldots, x_{2r}]$ is the **search pattern** of the r-round characteristic. Because the number of active S-boxes in an SB layer is fixed by either its input or its output state, the core trail corresponds to the search pattern $[x_2, \ldots, x_{2r-1}]$, while the extended trail corresponds to $[x_1, \ldots, x_{2r}]$. In our approach, we first find $(r-1)$-round core trails, then extend them to r-round extended trails, from which the r-round characteristics are finally derived.

Motivation for the core trail. The key innovation and the efficiency of our search both come from the core trail. Since both the minimum number of active S-boxes and the best differential characteristic can be obtained directly from the r-round extended trails, it

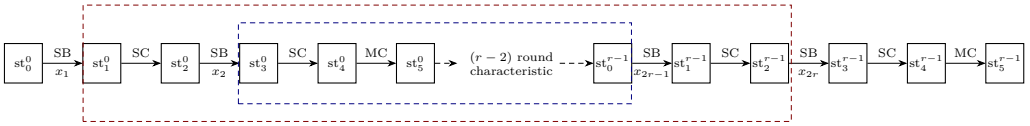

Figure 3: Decomposition of an r-round differential characteristic. The blue box is an $(r-1)$-round core trail, the red box an r-round extended trail, and the whole path an r-round characteristic.

suffices to enumerate these trails efficiently. Building an r-round extended trail from an $(r-1)$-round extended trail requires a single-side extension: a right-side extension models the path $\mathrm{st}_2^{r-2} \xrightarrow{\mathrm{SB}} \mathrm{st}_3^{r-2} \xrightarrow{\mathrm{SC}} \mathrm{st}_4^{r-2} \xrightarrow{\mathrm{MC}} \mathrm{st}_5^{r-2} \xrightarrow{\mathrm{SB}} \mathrm{st}_1^{r-1} \xrightarrow{\mathrm{SC}} \mathrm{st}_2^{r-1}$, or a left-side extension the reverse path $\mathrm{st}_1^0 \xleftarrow{\mathrm{SC}} \mathrm{st}_2^0 \xleftarrow{\mathrm{SB}} \mathrm{st}_3^0 \xleftarrow{\mathrm{SC}} \mathrm{st}_4^0 \xleftarrow{\mathrm{MC}} \mathrm{st}_5^0 \xleftarrow{\mathrm{SB}} \mathrm{st}_1^1$. Either extension contains two SB layers, making the model large and slow to solve. To avoid this, we introduce the core trail: an $(r-1)$-round core trail is extended to an r-round extended trail via two paths, $\mathrm{st}_1^0 \xleftarrow{\mathrm{SC}} \mathrm{st}_2^0 \xleftarrow{\mathrm{SB}} \mathrm{st}_3^0$ on the left and $\mathrm{st}_0^{r-1} \xrightarrow{\mathrm{SB}} \mathrm{st}_1^{r-1} \xrightarrow{\mathrm{SC}} \mathrm{st}_2^{r-1}$ on the right. Each of these two paths contains only a single SB layer, so the corresponding MILP models can be solved efficiently. Moreover, the two paths are independent and can be searched in parallel.

Search procedure. Let n be the total number of active S-boxes in an r-round extended trail, split as $n = n_1 + n_2$, where n_1 counts the active S-boxes in the internal $(r-1)$-round core trail and $n_2 = x_1 + x_{2r}$ counts those in the two outer layers. For a fixed n, Algorithm 1 generates all candidate r-round extended patterns, and Algorithm 2 searches for all extended trails conforming to each pattern. As n increases, the numbers of candidate patterns and corresponding trails grow rapidly, making the search too expensive in time and memory. We therefore impose the following constraint on every internal 1-round core, and let a be the threshold from which this constraint takes effect:

$$x_i + x_{i+1} \leq 14, \qquad i = 2, 4, \ldots, 2r-2.$$

This constraint may exclude some candidate patterns, so the search is not guaranteed to be exhaustive for $n \geq a$. For $n < a$, the search space is unaffected by the constraint and is explored exhaustively. This is the only restriction imposed throughout the search.

Let x be the minimum number of active S-boxes in the $(r-1)$-round core trails, as determined in Section 3.2. Since each of the two outer layers contains at least one active S-box, an extended trail has at least $x+2$ active S-boxes. To find the minimum number of active S-boxes of r-round `SPEEDY`, we sweep n upward from $x+2$, generating all candidate patterns with Algorithm 1 and searching them with Algorithm 2, until the first solution appears at $n = y$. If $y \leq a$, then all totals below y have been searched exhaustively, so $\#AS_{\min} = y$ is exact. Otherwise, the constraint may have excluded valid trails with $a \leq n < y$, and we can only conclude that $\#AS_{\min}$ lies in $[a, y]$. To obtain upper bounds on the differential probability, we continue searching over $n = y, y+1, y+2$ (limited by search time, we stop at $y+2$) and use the DDT to determine the best probability P_{BEST} that can be obtained from these extended trails. Let P_{UB} be the upper bound on the differential probability of the r-round characteristic. If $y+2 < a$, then $P_{\mathrm{UB}} = \max\{2^{-3(y+3)},\ P_{\mathrm{BEST}}\}$; otherwise, $P_{\mathrm{UB}} = \max\{2^{-3a},\ P_{\mathrm{BEST}}\}$. The whole procedure is summarized in Algorithm 3.

Algorithm 1 Generation of r-round extended patterns for a given n

Require: Number of rounds $r \in \{2, 3, 4, 5\}$, total active S-boxes n

Ensure: Table $T^{\text{potentially valid}}_{r\text{-round extended}}$ containing candidate r-round extended patterns with total active S-boxes n

Let the minimum number of active S-boxes in $(r-1)$-round core trails be x. ▷ The determination of x is processed in Section 3.2.

Phase 1: Generate candidate $(r-1)$-round core patterns

for $n_1 = x, x+1, \ldots, n-2$ **do**

  Generate all $(r-1)$-round core patterns $[x_2, \ldots, x_{2r-1}]$ with total n_1

  Filter out illegal patterns and store the remaining in $T^{\text{potentially valid}}_{(r-1)\text{-round core}}$

end for ▷ The detailed process is given in Section 3.2.

Phase 2: Generate candidate r-round extended patterns

Decompose n into all combinations $n = n_1 + n_2$ with $x \le n_1 \le n-2$ ▷ $coms_1$

Decompose n_2 into all combinations $n_2 = x_1 + x_{2r}$ ▷ $coms_2$

Generate all patterns $[x_1, \ldots, x_{2r}]$ from $coms_1$, $coms_2$, and $T^{\text{potentially valid}}_{(r-1)\text{-round core}}$

Filter out illegal patterns and store remaining in $T^{\text{potentially valid}}_{r\text{-round extended}}$

return $T^{\text{potentially valid}}_{r\text{-round extended}}$

Algorithm 2 Search for an r-round extended pattern

Require: An r-round extended pattern $[x_1, \ldots, x_{2r}]$

Ensure: The set E of all r-round extended trails corresponding to the given pattern.

$E \leftarrow \varnothing$

Search for the internal core pattern $[x_2, \ldots, x_{2r-1}]$ and denote the solution set as S_1 ▷ The detailed process is given in Section 3.2.

if $S_1 = \varnothing$ **then**

  Update the corresponding validity tables

  return $\varnothing$

end if

for all $t \in S_1$ **do**

  Extend t on both sides:

    L: fix st_3^0, search $\mathrm{st}_1^0 \xleftarrow{\mathrm{SC}} \mathrm{st}_2^0 \xleftarrow{\mathrm{SB}} \mathrm{st}_3^0$ that st_1^0 has x_1 active S-boxes

    R: fix st_0^{r-1}, search $\mathrm{st}_0^{r-1} \xrightarrow{\mathrm{SB}} \mathrm{st}_1^{r-1} \xrightarrow{\mathrm{SC}} \mathrm{st}_2^{r-1}$ that st_2^{r-1} has x_{2r} active S-boxes

  Concatenate the two extended parts with t and add the obtained trails to E

end for

return E

Algorithm 3 Security analysis procedure for r-round SPEEDY characteristics

Require: Number of rounds $r \in \{2, 3, 4, 5\}$
Ensure: $\#AS_{\min}$ (an exact value or an interval), P_{BEST}, and P_{UB}

Let x be the minimum number of active S-boxes in $(r-1)$-round core trails
Let a be the number from which the search space of r-round extended trails is restricted
Problem 1: Determine the minimum active S-box count $\#AS_{\min}$
for $n = x+2, x+3, \ldots$ **do**
    Generate all candidate r-round extended patterns with total n using Algorithm 1
    for all r-round extended patterns with total n **do**
        Apply Algorithm 2 to search for this pattern
        if a solution is found **then**
            $y \leftarrow n$
            exit both loops
        end if
    end for
end for
if $y \leq a$ **then**
    $\#AS_{\min} \leftarrow y$ ▷ exact value
else
    $\#AS_{\min} \leftarrow [a,\ y]$ ▷ an interval
end if
Problem 2: Determine the upper bound P_{UB} on differential probability
for $n = y, y+1, y+2$ **do**
    Generate all candidate r-round extended patterns with total n using Algorithm 1
    for all r-round extended patterns with total n **do**
        Apply Algorithm 2 to search for this pattern
        Add the obtained trails to the set of extended trails for probability evaluation
    end for
end for
Compute P_{BEST} from these extended trails using the DDT
if $a > y+2$ **then**
    $P_{\text{UB}} \leftarrow \max\{2^{-3(y+3)},\ P_{\text{BEST}}\}$
else
    $P_{\text{UB}} = \max\{2^{-3a},\ P_{\text{BEST}}\}$
end if
return $\#AS_{\min}$, P_{BEST}, and P_{UB}

3.2 Efficient search for r-round core Trails

The efficiency of Algorithm 2 relies on enumerating the core trails and extending them. This section introduces an efficient method to search for r-round core trails and to determine their minimum number of active S-boxes, for $r = 1, 2, 3, 4$.

Short-round reuse via hash-map concatenation. An MILP model can be built with OCP to search for r-round core trails directly. However, this model grows large and slow as r increases. We therefore build MILP models only for the short cores: the 1-round cores directly, and the 2-round cores by joining two 1-round cores through another MILP model. The long cores ($r = 3, 4$) are then constructed by concatenating 2-round cores that share an intermediate state, matched efficiently through a hash-map keyed by that state. This concatenation is extremely cheap and requires no MILP model, so the cost concentrates on the 1- and 2-round searches. Moreover, since both 3- and 4-round cores are built from 2-round cores, the 2-round cores are computed once and reused, saving further time.

A general search procedure for r-round core trails. First, for a fixed total number n of active S-boxes, we generate all the candidate search patterns $[x_2, \ldots, x_{2r+1}]$ by decomposing n using the already-computed shorter cores. Then, we use some simple checking conditions or low-cost searches to preliminarily check the validity of these search patterns, thereby avoiding many unnecessary searches. Finally, we search for r-round core trails in increasing order of n, and determine their minimum number of active S-boxes. The schematic diagrams of 1-, 2-, 3-, and 4-round core trails are shown in Figure 4, where x_i denotes the number of active S-boxes in the corresponding state.

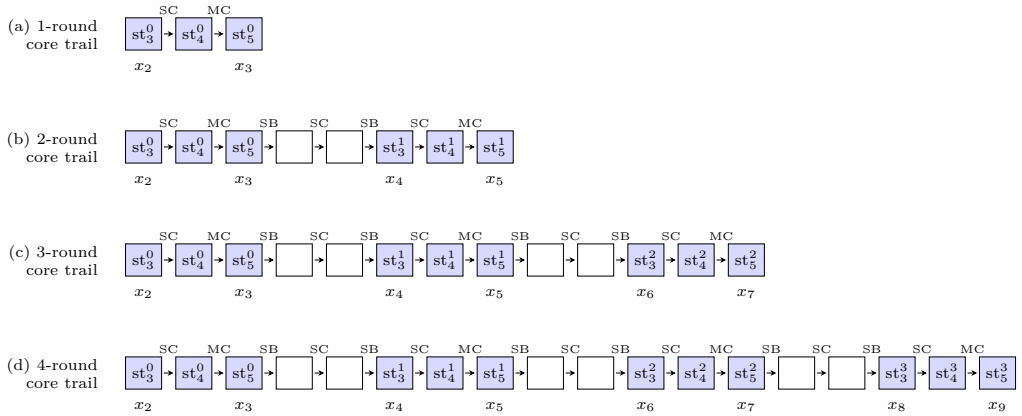

Figure 4: 1-, 2-, 3-, and 4-round core trails.

Step 1: Pattern generation. The 1-round cores form the base case, and its pattern generation is straightforward: we simply decompose n into all combinations $n = x_2 + x_3$. To generate multi-round core patterns, we first perform a preliminary validity check on the 1-round core patterns: model and check whether the path $\mathrm{st}_3^0 \xrightarrow{\mathrm{SC}} \mathrm{st}_4^0 \xrightarrow{\mathrm{MC}} \mathrm{st}_5^0$ has a solution under the constraint that st_3^0 and st_5^0 have x_2 and x_3 active S-boxes, respectively. This model is very fast to solve, since the solver stops as soon as a feasible solution is found. Patterns with a solution are written into $T^{\text{valid}}_{\text{1-round core}}$, while those without a solution are written into $T^{\text{invalid}}_{\text{1-round core}}$. On one hand, since the branch number of the MC layer is 8, the minimum number of active S-boxes in 1-round core trails is 8. On the other hand, we impose the constraint that, in each of the first $r-1$ rounds, the total number of active S-boxes in st_3 and st_5 does not exceed 14. Therefore, the validity tables above record all 1-round core patterns for n ranging from 8 to 14. For r-round core trails with $r = 2, 3, 4$, they can be viewed as r 1-round core trails connected by $r-1$ intermediate $\mathrm{SB} \circ \mathrm{SC} \circ \mathrm{SB}$ layers. First, we decompose n into all combinations of r positive integers, i.e., $n = n_1 + \cdots + n_r$, where $8 \le n_i \le 14$. Then, using $T^{\text{valid}}_{\text{1-round core}}$, we decompose each n_i into $x_{2i} + x_{2i+1}$. Finally, we obtain the r-round core patterns $[x_2, x_3, \ldots, x_{2r+1}]$.

Step 2: Validity checking. The validity checking is mainly based on checking the active S-box counts and the validity of inner subpatterns. Specifically, an r-round core pattern $[x_2, \ldots, x_{2r+1}]$ is discarded if it violates either of the following:

- A consecutive k-round core subpattern contains fewer than the minimum number of active S-boxes established for k rounds, for $2 \le k < r$.
- An embedded subpattern $[x_i, \ldots, x_j] \subsetneq [x_2, \ldots, x_{2r+1}]$ is shown to be invalid.

The minimum numbers of S-boxes for r-round core trails are shown in Table 3. For any 2-round core subpattern, the number of active S-boxes is at least 20; for any 3-round core subpattern, it is at least 30.

For embedded subpatterns, we only consider the consecutive k-round core subpatterns with $2 \leq k < r$ and the 3-term connection subpatterns $[x_i, x_{i+1}, x_{i+2}]$ for $2 \leq i \leq 2r-1$. The validity check of the k-round core subpatterns is performed by looking up the corresponding validity table $T^{\text{invalid}}_{k\text{-round core}}$. The validity check of the 3-term connection subpatterns is performed by extending the corresponding 1-round core trails. For example, to check $[x_2, x_3, x_4]$, we first enumerate the 1-round core trails with pattern $[x_2, x_3]$. For each such trail, we fix its final state st_5^0 and search for a propagation $\text{st}_5^0 \rightarrow \text{SB}\circ\text{SC}\circ\text{SB} \rightarrow \text{st}_3^1$ such that st_3^1 contains exactly x_4 active S-boxes. The connection subpattern is valid if at least one such propagation exists. The connection subpattern $[x_3, x_4, x_5]$ can be checked analogously by extending 1-round core trails with pattern $[x_4, x_5]$ backward.

Finally, illegal patterns are stored in $T^{\text{invalid}}_{r\text{-round core}}$, while the remaining patterns are stored in $T^{\text{potentially valid}}_{r\text{-round core}}$.

Step 3: Pattern search. Let S_P denote the set of trails corresponding to a core pattern P.

- *1-round* $[x_2, x_3]$: Model and solve $\text{st}_3^0 \xrightarrow{\text{SC}} \text{st}_4^0 \xrightarrow{\text{MC}} \text{st}_5^0$ with x_2 and x_3 active S-boxes in st_3^0 and st_5^0, respectively. By the translation invariance of the `SPEEDY` round functions (a cyclic shift of all rows maps one trail to another), we keep one representative per cyclic-shift class.

- *2-round* $[x_2, x_3, x_4, x_5]$: The search for 2-round core patterns is based on 1-round core patterns. For each pair of 1-round core trails from $S_{[x_2,x_3]}$ and $S_{[x_4,x_5]}$, we fix their boundary states st_5^0 and st_3^1, build an MILP model to search for the connecting path $\text{st}_5^0 \xrightarrow{\text{SB}\circ\text{SC}\circ\text{SB}} \text{st}_3^1$, then concatenate the intermediate path with the two 1-round core trails to obtain 2-round core trails. Although this model spans two SB layers, fixing both ends makes it very fast to solve. To avoid missing solutions, one side is matched over all remaining 31 cyclic shifts. This process requires approximately $|S_{[x_2,x_3]}| \times |S_{[x_4,x_5]}| \times 32$ modeling and search operations, which can be parallelized.

- *3-round* $[x_2, x_3, x_4, x_5, x_6, x_7]$: A 3-round core trail is obtained by concatenating the two 2-round core trails from $S_{[x_2,x_3,x_4,x_5]}$ and $S_{[x_4,x_5,x_6,x_7]}$ that share the middle 1-round core trail $[x_4, x_5]$. We hash $S_{[x_2,x_3,x_4,x_5]}$ and $S_{[x_4,x_5,x_6,x_7]}$ into map_1 and map_2, respectively, using the shared 1-round core $[x_4, x_5]$ as the key and storing the corresponding trails as values. Then we traverse the two hash maps and concatenate a 2-round core trail from map_1 with a 2-round core trail from map_2 that share the same key, thereby obtaining a 3-round core trail. Note that during the traversal, we need to perform the remaining 31 nontrivial cyclic shifts on the keys of one of the two hash maps (and apply the same cyclic shift to the corresponding 2-round core trails).

- *4-round* $[x_2, x_3, x_4, x_5, x_6, x_7, x_8, x_9]$: A 4-round core trail is obtained by concatenating the three 2-round core trails from $S_{[x_2,x_3,x_4,x_5]}$, $S_{[x_4,x_5,x_6,x_7]}$, and $S_{[x_6,x_7,x_8,x_9]}$, where adjacent 2-round cores share the middle 1-round cores $[x_4, x_5]$ and $[x_6, x_7]$. We hash $S_{[x_2,x_3,x_4,x_5]}$ and $S_{[x_6,x_7,x_8,x_9]}$ into map_1 and map_2, keyed by the shared 1-round cores $[x_4, x_5]$ and $[x_6, x_7]$, respectively, with the corresponding trails stored as values. Then we traverse $S_{[x_4,x_5,x_6,x_7]}$ and check whether the 1-round core $[x_4, x_5]$ (with the remaining 31 nontrivial cyclic shifts) of the current trail exists in map_1, and whether the 1-round core $[x_6, x_7]$ (with the remaining 31 nontrivial cyclic shifts) exists in map_2. If both exist, we concatenate these corresponding 2-round core trails (after applying the corresponding cyclic shift) to obtain 4-round core trails.

Minimum number of active S-boxes in r-round core trails. We obtain the minimum number of active S-boxes for r-round core trails, by searching for the core patterns in

increasing order of n. The starting value of n is determined based on the results of the shorter rounds. For example, for 3-round core trails, since the minimum numbers of active S-boxes in 1-round and 2-round core trails are 8 and 20, respectively, the search starts from $n = 28$. The search stops when a solution is found with $n = x$. Let b denotes the value of n from which the search space is restricted due to the constraint that any 1-round core trail has at most 14 active S-boxes. If $x \leq b$, the search up to x is exhaustive, so the minimum number of active S-boxes in r-round core trails is exactly x; otherwise, it lies in the interval $[b, x]$. The results are summarized in Table 3.

Table 3: The minimum number of active S-boxes for r-round core trails.

Rounds	Start n	First solution at $n = x$	Search restricted from $n = b$	$\#AS_{\min}$
1	8	8	15	8
2	16	20	23	20
3	28	30	31	30
4	40	42	43	42

3.3 New security bounds of SPEEDY against differential cryptanalysis

In this section, we apply Algorithm 3 to re-evaluate the minimum number of active S-boxes and the upper bound on the differential probability of r-round SPEEDY ($r = 2, 3, 4, 5$), improving on the security bounds against differential cryptanalysis.

Security evaluation results. Recall from Table 3, the minimum numbers of active S-boxes in 1-, 2-, 3-, and 4-round core trails are $x = 8, 20, 30, 42$, respectively; and each search space becomes restricted from the thresholds $b = 15, 23, 31, 43$, respectively. An r-round extended trail is obtained by extending an $(r-1)$-round core trail with two outer layers, so its search space becomes restricted as soon as the inner core does, namely from $a = 17, 25, 33, 45$ for $r = 2, 3, 4, 5$, respectively. Algorithm 3 then returns the security evaluation results:

- **2 rounds** ($x = 8$, $a = 17$): $\#AS_{\min} = 13$, $P_{\text{BEST}} = 2^{-46.245}$, and $P_{\text{UB}} = 2^{-46.245}$.
- **3 rounds** ($x = 20$, $a = 25$), $\#AS_{\min} = 23$, $P_{\text{BEST}} = 2^{-76.15}$, and $P_{\text{UB}} = 2^{-75}$.
- **4 rounds** ($x = 30$, $a = 33$), $\#AS_{\min} \in [33, 34]$, $P_{\text{BEST}} = 2^{-119.735}$, and $P_{\text{UB}} = 2^{-99}$.
- **5 rounds** ($x = 42$, $a = 45$), $\#AS_{\min} = 45$, $P_{\text{BEST}} = 2^{-156.565}$, and $P_{\text{UB}} = 2^{-135}$.

The corresponding differential characteristics with the minimum number of active S-boxes are presented in Appendix C.1; and the corresponding differential characteristics with the best differential probability are presented in Appendix C.2.

Comparison and summary of security bounds. We now compare our security results with those of the designers [LMMR21] and of Zhang [Zha24] for SPEEDY across 2 to 5 rounds. The results of both the designers and Zhang were obtained under certain restrictions, such as 1-bit-to-1-bit differential propagation in some SB layers, whereas our results are obtained in the global search space. Table 4 compares the minimum numbers of active S-boxes. For 4 rounds, both the designers and Zhang gave a lower bound of 35, whereas we determine that the minimum number of active S-boxes lies in the interval $[33, 34]$. In particular, we found a 4-round differential characteristic with only 34 active S-boxes that no earlier framework can cover, because in the first SB layer of its second round there is a 1-bit-to-multi-bit differential propagation (see Table 16 in Appendix C.1). Table 5 compares the differential probability bounds, where P_{BEST} denotes the best result found in the corresponding search,

and P_{UB} denotes the estimated upper bound on the differential probability. Zhang's results disprove the designers' upper bounds on the differential probability (branch-number-based security evaluation) for 2 and 3 rounds, while our results disprove them for 2, 3, and 4 rounds. The probabilities we find are the best known to date. In addition, we exhaustively searched all 2-round differential characteristics with 13, 14, and 15 active S-boxes, and all 3-round differential characteristics with 23 and 24 active S-boxes, so our 2- and 3-round bounds do not merely follow from the minimum active-S-box count. In particular, the 2-round probability $P_{\mathrm{BEST}} = 2^{-46.245}$ is globally optimal.

Table 4: Comparison of the minimum number of active S-boxes ($\#AS_{\min}$) for SPEEDY.

Rounds	$\#AS^*_{\min}$ [LMMR21]	$\#AS^*_{\min}$ [Zha24]	$\#AS_{\min}$ in Sect. 3.3
2	13	13	13
3	23	23	23
4	35	35	[33, 34]
5	–	45	45

$\#AS^*_{\min}$ denotes the constrained evaluation, while $\#AS_{\min}$ denotes the global estimate.

Table 5: Comparison of the differential probability bounds for SPEEDY, where P_{BEST} denotes the best result found in the corresponding search, and P_{UB} denotes the estimated upper bound on the differential probability.

	Designers [LMMR21]	Zhang [Zha24]		Sect. 3.3	
Rounds	P^*_{UB}	P_{BEST}	P^*_{UB}	P_{BEST}	P_{UB}
2	2^{-48}	$2^{-46.2}$	2^{-39}	$2^{-46.245}$	$2^{-46.245}$
3	2^{-78}	$2^{-76.72}$	2^{-69}	$2^{-76.15}$	2^{-75}
4	2^{-120}	$2^{-129.2}$	2^{-105}	$2^{-119.735}$	2^{-99}
5	2^{-150}	2^{-170}	2^{-135}	$2^{-156.565}$	2^{-135}

P^*_{UB} denotes the constrained evaluation, while P_{UB} denotes the global estimate.

For 2 rounds, Zhang's $2^{-46.2}$ and our $2^{-46.245}$ are essentially the same value up to rounding.

Comparison of the search spaces of [BDG+25b] and our work. In [BDG+25b], Boura et al. build r-round differential characteristics by concatenating one-round characteristics; each one-round characteristic required to satisfy one of the following constraints:

(1) $T = \{x \in \mathbb{F}_2^{192} \mid w(\mathrm{SC}^{-1} \circ \mathrm{MC}^{-1}(x)) + w(x) \leq 12\}$, $\mathrm{st}_0, \mathrm{st}_5 \in T$, $p > 2^{-60}$;

(2) $T' = \{x \in \mathbb{F}_2^{192} \mid w(\mathrm{SC}^{-1} \circ \mathrm{MC}^{-1}(x)) + w(x) \leq 14\}$, $\mathrm{st}_0, \mathrm{st}_5 \in T'$, $p > 2^{-46}$;

(3) $T'' = \{x \in \mathbb{F}_2^{192} \mid w(\mathrm{SC}^{-1} \circ \mathrm{MC}^{-1}(x)) + w(x) \leq 16\}$, $\mathrm{st}_0, \mathrm{st}_5 \in T''$, $p > 2^{-25}$.

Here, st_0 and st_5 denote the initial and final differential states of a one-round differential characteristic, respectively; $w(x)$ denotes the number of active S-boxes in the state x, and p denotes the probability of a one-round differential characteristic.

During our search, the restriction that any 1-round core trail contains at most 14 active S-boxes is equivalent to a similar restriction:

$$T' = \{x \in \mathbb{F}_2^{192} \mid w(\mathrm{SC}^{-1} \circ \mathrm{MC}^{-1}(x)) + w(x) \leq 14\}, \quad \mathrm{st}_0, \mathrm{st}_5 \in T'.$$

However, it is worth noting that we only impose this restriction on the internal st_0 and st_5 of the r-round characteristic, but not on the starting difference (first round's st_0^0) and the ending difference (last round's st_5^{r-1}) of the r-round characteristic. Moreover, throughout

our entire search for differential characteristics, we do not impose any restriction on the probability of the characteristic. Consequently, for a fixed total number of active S-boxes in an r-round characteristic, our search space covers the search spaces corresponding to the above cases (1) and (2) in [BDG+25b]. In addition, we also explore an extra part of the search space, namely those r-round characteristics whose initial difference (first round's st_0^0) and final difference (last round's st_5^{r-1}) do not satisfy $\mathrm{st}_0, \mathrm{st}_5 \in T'$. Concretely, when the first SB layer of a characteristic has only one active S-box, i.e., the initial difference st_0^0 has only one active S-box, then due to the strong diffusion of MC^{-1} (if a column of the output state y of the MC layer has only one active bit, then applying MC^{-1} gives the corresponding column of the input state x with 19 active bits), we have $w(\mathrm{SC}^{-1} \circ \mathrm{MC}^{-1}(x)) \geq 19$, so $\mathrm{st}_0^0 \notin T, T', T''$. In other words, all differential characteristics found in [BDG+25b] have at least two active S-boxes in the first SB layer. In fact, it is precisely in this extra search space that we find the above best known differential characteristics.

For key recovery attacks, we need the complete r-round differential characteristics, obtained as described in Section 3.1. We then verify them and compute their expected differential probability (EDP) with the quasidifferential framework from [BDG25a]. For every $r = 2, 3, 4, 5$, we improve the best previously known EDPs from [BDG+25b], obtaining the best values known to date (Table 6). The characteristics achieving these EDPs also lie in the extra region identified above, outside the search space of [BDG+25b]. The corresponding differential characteristics are presented in Appendix C.3.

Table 6: Comparison of the best expected differential probabilities for r-round SPEEDY.

Rounds	Boura et al. [BDG+25b]	Ours
2	$2^{-49.24}$	$2^{-46.245}$
3	$2^{-78.56}$	$2^{-74.998}$
4	$2^{-120.58}$	$2^{-117.32}$
5	$2^{-153.9}$	$2^{-152.565}$

4 Key recovery on SPEEDY-7-192

In this section, we mount key recovery attacks on SPEEDY-7-192 using 5-round distinguishers. The enlarged search of Section 3.3 refines the security evaluation of SPEEDY from a design perspective by finding stronger 5-round characteristics. For key recovery, however, the relevant objective is the full data-time-memory tradeoff after probabilistic extension, rather than the middle distinguisher probability alone. Accordingly, our best attack reuses the same middle distinguisher and preserves almost the same memory complexity as the attack of [BDG+25b], but reduces both the data and time complexities by optimizing the probabilistic extension, setting $l_{\mathrm{in}} + l_{\mathrm{out}} = 31$ instead of 32.

4.1 Optimized probabilistic extension

We cluster the 5-round characteristics with 45 or 46 active S-boxes and expected differential probabilities above 2^{-170}, i.e., grouping those with identical input and output differences into a single distinguisher. Each then serves as the middle distinguisher of the 7-round attack, and we optimize its probabilistic extension to minimize the overall cost.

Cost model. For each 5-round middle distinguisher, the plaintext-side and ciphertext-side probabilistic extensions are determined before mounting the 7-round key recovery attack. Let 2^{-p} be the probability of the 5-round middle distinguisher, and let $2^{-p_{\mathrm{in}}}$ and $2^{-p_{\mathrm{out}}}$

be the probabilities of the plaintext-side and ciphertext-side probabilistic extensions. The probability of the extended differential is then denoted by $2^{-p'}$, where

$$p' = p_{\text{in}} + p + p_{\text{out}}.$$

As known from Section 2.2, the data generation phase requires about $2^{p'+1}$ data complexity. Let l_{in} and l_{out} be the numbers of inactive rows on the plaintext side and the ciphertext side after the extension, respectively. Then the expected number of message pairs entering key recovery phase is

$$2^{p'+192-6\cdot(l_{\text{in}}+l_{\text{out}})}.$$

For a fixed distinguisher, a smaller l_{in} (l_{out}) generally gives a smaller p_{in} (p_{out}). On the one hand, reducing the value of $l_{\text{in}} + l_{\text{out}}$ decreases p' and thus lowers the data and time complexities of the data generation phase; however, it also increases the number of message pairs entering the key recovery phase, thereby raising the time complexity of that phase. On the other hand, the choices of l_{in} and l_{out} separately affect the memory complexity under chosen-plaintext and chosen-ciphertext attacks, respectively. Hence, the extension parameters have to be selected according to the resulting data-time-memory tradeoff.

Optimizing $l_{\text{in}} + l_{\text{out}}$. In [BDG+25b], the authors fix $l_{\text{in}} + l_{\text{out}}$ to 32 and minimize p' in the probabilistic extension. About $2^{p'}$ message pairs then enter key recovery, and since their key recovery strategy applies only simple operations to these pairs, its cost is about $2^{p'}$ simple operations, negligible against the data generation phase ($2^{p'+1}$ encryptions). Hence, the overall time complexity is approximately equal to that of the data generation phase. We consider reducing the value of $l_{\text{in}} + l_{\text{out}}$, which lowers the time complexity of the data generation phase while increasing the time complexity of the key recovery phase. If the time complexity of the key recovery phase remains negligible or accounts for only a small proportion compared to the data generation phase, then the overall time complexity will decrease. When $l_{\text{in}} + l_{\text{out}}$ is reduced from 32 to 31, the number of message pairs entering the key recovery phase increases from $2^{p'}$ to $2^{p'+6}$. Due to the efficiency of the key recovery strategy, the time complexity of key recovery phase is still not dominant, so the total time complexity decreases. However, when $l_{\text{in}} + l_{\text{out}}$ is further reduced to 30, the number of message pairs entering key recovery increases to $2^{p'+12}$. In this scenario, the time complexity of the key recovery phase becomes the dominant part of the attack, and the total time complexity increases. Hence, among the evaluated choices, $l_{\text{in}} + l_{\text{out}} = 31$ gives the best overall tradeoff.

Effect on memory. The distribution between l_{in} and l_{out} will affect the memory complexity. We ignore the additional memory overhead incurred during the key recovery phase. In a chosen-ciphertext attack, when *structures* are used for decryption, the memory complexity is $2^{6\cdot(32-l_{\text{out}})}$ complete states. When applying the probabilistic extension to the same distinguisher with $l_{\text{in}} + l_{\text{out}} = 31$ fixed, different combinations of $(l_{\text{in}}, l_{\text{out}})$ may yield similar values of p' (and thus similar data and time complexities), but the memory complexity can vary significantly with different choices of l_{out}. This is why we compare candidate attacks not only by p', but also by l_{out}.

Candidate distinguishers. We performed the above optimized probabilistic extension for the distinguishers whose expected differential probability exceeds 2^{-162}. This cutoff is used to focus the extension search on candidates that may improve the best existing attack. For each candidate distinguisher, the plaintext-side and ciphertext-side extensions can be performed independently. Fixing l_{in} and l_{out}, the value p_{in} depends only on the input difference of the middle distinguisher, while p_{out} depends only on its output difference. We therefore first extract the distinct input and output differences from the candidate set,

compute the best extensions for both sides, and then combine the two extension costs with the middle weight p to obtain p'.

Some distinguishers found in the enlarged search space have higher probabilities than the middle distinguisher used in [BDG+25b], with the highest probability being $2^{-151.766}$. However, their boundary differences may lead to more expensive probabilistic extensions. Specifically, the starting differences st_0^0 of the 5-round distinguishers obtained in the enlarged search space usually has a single active S-box. After applying $\mathrm{SC}^{-1} \circ \mathrm{MC}^{-1}$ to such an st_0^0, the resulting state (denoted by Δ_{in}) has at least 19 active S-boxes (whereas at most 14 in [BDG+25b]). The probabilistic extension on the plaintext-side is performed on Δ_{in}. Consequently, the cost p_{in} of the plaintext-side probabilistic extension becomes prohibitively large, leading to a large overall p' after the probabilistic extension, which is unsuitable for a key recovery attack.

Table 7 summarizes the attack on SPEEDY-7-192 in [BDG+25b] together with our two leading candidate attacks after probabilistic extension. Coincidentally, the 5-round distinguishers[4] corresponding to these two candidate attacks are in the same cyclic shift equivalence class as the one used in [BDG+25b] (i.e., applying the same cyclic shift to both the input and output differences of the distinguisher yields the other), only with different probabilistic extension parameters $(l_{\mathrm{in}}, l_{\mathrm{out}})$. Candidate 1 gives the smallest value of p', hence the lowest data and time complexities, but requires 2^{72} memory complexity, which is a factor of 2^6 more than that in [BDG+25b]. Candidate 2 has a slightly higher p' (i.e., it is slightly worse in terms of data and time complexites), but the memory complexity is only $2^{66.06}$. We describe the key recovery procedure of Candidate 2 in Section 4.2. The data for Candidate 1 are given in Appendix F, and its attack follows the similar procedure.

Table 7: Comparison of the attack on SPEEDY-7-192 in [BDG+25b] and the two candidate attacks in our work. Here, 2^{-p} is the distinguisher probability, l_{in} and l_{out} are the inactive-row counts after extension, $2^{-p_{\mathrm{in}}}$ and $2^{-p_{\mathrm{out}}}$ are the extension probabilities, and $2^{-p'}$ is the total extended probability.

Attack	2^{-p}	$(l_{\mathrm{in}}, l_{\mathrm{out}})$	$(2^{-p_{\mathrm{in}}}, 2^{-p_{\mathrm{out}}})$	$2^{-p'}$	Data	Time	Memory
[BDG+25b]	$2^{-158.36}$	$(11, 21)$	$(2^{-6.17}, 2^{-9})$	$2^{-173.53}$	$2^{174.53}$	$2^{174.53}$	2^{66}
Candidate 1	$2^{-158.36}$	$(11, 20)$	$(2^{-6.17}, 2^{-7.95})$	$2^{-172.48}$	$2^{173.48}$	$2^{173.69}$	2^{72}
Candidate 2	$2^{-158.36}$	$(10, 21)$	$(2^{-5.19}, 2^{-9})$	$2^{-172.55}$	$2^{173.55}$	$2^{173.76}$	$2^{66.06}$

4.2 Key recovery

The final determined distinguisher (the corresponding dominant differential trails can be found in Appendix D) and its corresponding probabilistic extension are shown in Figure 5, where $(l_{\mathrm{in}}, l_{\mathrm{out}}) = (10, 21)$, and $p' = p_{\mathrm{in}} + p + p_{\mathrm{out}} = 5.19 + 158.36 + 9.00 = 172.55$. According to Section 2.2, during the data generation phase, an average of $2^{173.55}$ messages need to be chosen to produce one pair that satisfies the entire differential propagation path, and the number of message pairs entering the key recovery phase is about $2^{178.55}$. The entire key recovery procedure we adopt is broadly the same as the one used by Boura et al. in their key recovery attack on SPEEDY-7-192 [BDG+25b]. For convenience, we name the two SB layers on the plaintext side as SB_1 and SB_2, and the two SB layers on the ciphertext side as SB_{13} and SB_{14}, respectively. The states before and after each of these four SB layers are denoted as st_0^m, st_1^m, st_2^m, st_3^m on the plaintext side, and st_0^c, st_1^c, st_2^c, st_3^c on the ciphertext side, respectively.

[4] Verified using the quasidifferential framework from [BR22], the candidate distinguisher is valid under any key, with its probability ranging from $2^{-158.39}$ to $2^{-158.34}$ [BDG+25b].

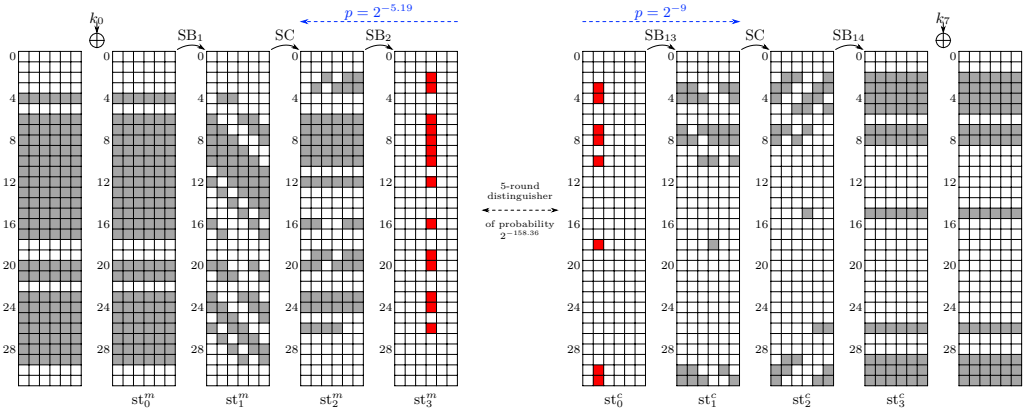

Figure 5: The final determined distinguisher and its corresponding probabilistic extension for the key recovery attack on SPEEDY-7-192. Red cells indicate a difference of 1, white cells indicate a difference of 0, and grey cells indicate bits with unknown difference.

Table 8: Filter values (in $\log_2$) of active rows in SB_1 and SB_{14}.

SB_1				SB_{14}	
Row	Filter	Row	Filter	Row	Filter
4	0.476	16	0.385	2	0.046
6	0.069	17	0.476	3	0
7	0.023	20	0.117	4	0.023
8	0	21	0.046	5	0.046
9	0	23	0.069	7	0.069
10	0	24	0.023	8	0.752
11	0	25	0.046	15	1.541
12	0	26	0.069	26	0.956
13	0	27	0.117	29	0.752
14	0.167	28	0.167	30	0.046
15	0.642	29	0.642	31	0.093
Total filter: 7.858					

Step 1. Filter the remaining message pairs based on the possibility of differential propagation in the active rows of SB_1 and SB_{14}. Specifically, we check whether the differential values in the active rows of the plaintext side can legally propagate through SB_1 to the differential states in the active rows of st_1^m; and whether the differential values in the active rows of the ciphertext side can legally propagate through SB_{14} to the differential states in the active rows of st_2^c. The filter values of the active rows in SB_1 and SB_{14} are shown in Table 8. For an active row, the filter value $2^{-filter}$ is the probability that a message pair passes the corresponding filtering condition. To minimize the time complexity of this step, we perform the filtering in descending order of the *filter* values. Firstly, for the remaining $2^{178.55}$ message pairs, we check whether the differential of row 15 on the ciphertext side is legal. We precompute the legality of each differential and record it in a one-dimensional list of length 2^6, where 1 indicates legal and 0 indicates illegal. This process requires $2^{178.55}$ table lookups, leaving $2^{178.55-1.54} = 2^{177.01}$ message pairs. Then, the remaining message pairs are filtered according to the other active rows, finally leaving $2^{178.55-7.86} = 2^{170.69}$ message pairs. This step involves a total of $2^{179.47}$ table lookups, where the time cost of one table lookup is approximately equal to that of one S-box lookup,

a full 7-round encryption includes 448 S-box lookups. Therefore, the time complexity of this step does not exceed $2^{179.47} \times 2^{-8.81} = 2^{170.66}$ encryptions. This step has the highest time complexity in the entire key recovery phase.

Step 2. Recover the key bits corresponding to the active rows on the plaintext side and the ciphertext side. The main idea of this step is to precompute and store all possible values of the active rows in st_0^m on the plaintext side and st_3^c on the ciphertext side, which is achieved through a back-and-forth procedure between the four involved SB layers. Specifically, for a given message pair, determining the values of certain rows in st_0^m and st_3^c reveals the corresponding key bits. By trading space for time, the online time complexity of this step is reduced. Table 9 records the number of possible values for the active rows in st_2^m and st_1^c. Key bits of the round key k_7 are shown in Table 25 in Appendix E.

Table 9: The number (in $\log_2$) of possible values for the active rows in st_2^m and st_1^c, i.e., each entry m corresponds to 2^m possible values.

st_2^m				st_1^c	
Row	m	Row	m	Row	m
2	5.09	16	5.17	3	5
3	5	19	5.39	4	5.46
6	6	20	5.46	7	5.58
7	6	23	6	8	5.58
8	6	24	6	10	4.58
9	6	26	4.7	18	3
10	6			30	4.32
12	6			31	5.46

(a) Recover the key bits corresponding to rows $[26, 29, 30, 31]$ in k_7 on the ciphertext side. The number of possible values for rows $30, 31$ in st_1^c is $2^{4.32+5.46}$. For rows $26, 29, 30, 31$ in st_2^c, there are still 3 bits of difference unknown, so the total number of possible values for these four rows in st_2^c is $2^{9.78+3+17} = 2^{29.78}$. The number of possible differences for rows $26, 29, 30, 31$ on the ciphertext side is $2^{24-\mathit{filter}}$, where $\mathit{filter} = 0.956+0.752+0.046+0.093$ is the filter value corresponding to the active rows in Step 1. Consequently, the average number of candidate value pairs per difference for rows $26, 29, 30, 31$ in st_3^c is $2^{29.78-24+1.85} = 2^{7.63}$. We precompute all $2^{29.78}$ possible value pairs for these four rows in st_3^c and store them in a hash table T indexed by the difference values. For each of the remaining $2^{170.69}$ message pairs, we compute the differences of rows $26, 29, 30, 31$ on the ciphertext side and look up T to recover the corresponding key bits in k_7 for these rows. After this step, the average number of remaining tuples $((P, C), (P', C'), k)$ is $2^{170.69+7.63} = 2^{178.32}$, and 24 key bits are determined.

(b) Recover the key bits corresponding to rows $[20, 21, 23\text{–}29]$ in k_0 on the plaintext side. The number of possible values for rows $19, 20, 23, 24, 26$ in st_2^m is $2^{27.55}$. For rows $20, 21, 23\text{–}29$ in st_1^m, there are still 2 bits of difference unknown, so the total number of possible values for these rows in st_1^m is $2^{27.55+2+29} = 2^{58.55}$. The number of possible differences for rows $20, 21, 23\text{–}29$ on the plaintext side is $2^{9\times6-1.3} = 2^{52.7}$. Consequently, the average number of candidate value pairs per difference for these rows in st_0^m is $2^{58.55-52.7} = 2^{5.85}$. Among the 54 key bits recovered in this step, 6 bits overlap with those recovered in step (a). Therefore, after this step, the average number of remaining tuples is $2^{178.32+5.85-6} = 2^{178.17}$, and a total of 72 key bits are determined.

(c) Recover the key bits corresponding to rows $[2, 3, 4, 5, 7, 8]$ in k_7 on the ciphertext side. The number of possible values for rows $3, 4, 7, 8, 10$ in st_1^c is $2^{26.2}$. For rows $2, 3, 4, 5, 7, 8$ in st_2^c, the number of possible values is $2^{26.2+18} = 2^{44.2}$. The number of possible differences for rows $2, 3, 4, 5, 7, 8$ on the ciphertext side is $2^{6\times6-0.94} = 2^{35.06}$. Consequently, the average number of candidate value pairs per difference for these rows in st_3^c is $2^{44.2-35.06} = 2^{9.14}$.

Note that 3 bits of values in rows $30, 31$ of st_2^c have already been determined in step (a); additionally, 12 key bits recovered in this step overlap with those recovered in step (b). Therefore, after this step, the average number of remaining tuples is $2^{178.17+9.14-2\times3-12} = 2^{169.31}$, and a total of 96 key bits are determined.

(d) Recover the key bits corresponding to rows [9–17] in k_0 on the plaintext side. The number of possible values for rows $7, 8, 9, 10, 12, 16$ in st_2^m is $2^{35.17}$. For rows 9–17 in st_1^m, the number of possible values is $2^{35.17+3+25} = 2^{63.17}$. The number of possible differences for rows 9–17 on the plaintext side is $2^{9\times6-1.67} = 2^{52.33}$. Consequently, the average number of candidate value pairs per difference for these rows in st_0^m is $2^{63.17-52.33} = 2^{10.84}$. Note that 2 bits of values in rows $20, 21$ of st_1^m have already been determined in step (b); additionally, 22 key bits recovered in this step overlap with those recovered in previous steps. Therefore, after this step, the average number of remaining tuples is $2^{169.31+10.84-2\times2-22} = 2^{154.15}$, and a total of 128 key bits are determined.

(e) Recover the key bit corresponding to row 15 in k_7 on the ciphertext side. The number of possible values for row 18 in st_1^c is 2^3. For row 15 in st_2^c, the number of possible values is $\min\{2^{3+5}, 2^{1+5}\} = 2^6$ (this is because the unknown difference of row 18 in st_1^c is necessarily 1, so the unknown difference of row 15 in st_2^c must also be 1). The number of possible differences for row 15 on the ciphertext side is $2^{6-1.54} = 2^{4.46}$. Consequently, the average number of candidate value pairs per difference for row 15 in st_3^c is $2^{6-4.46} = 2^{1.54}$. Additionally, 1 key bit recovered in this step overlaps with those recovered in previous steps. Moreover, the inactive rows $13, 14, 16, 17, 18$ in st_2^c contain bits related to row 18 of st_1^c, contributing an extra factor of $2^{(3-1)}$. Therefore, after this step, the average number of remaining tuples is $2^{154.15+1.54-1+(3-1)} = 2^{156.69}$, and a total of 133 key bits are determined.

(f) Recover the key bits corresponding to rows $[4, 6, 7, 8]$ in k_0 on the plaintext side. The number of possible values for rows $2, 3, 6$ in st_2^m is $2^{16.09}$. For rows $4, 6, 7, 8$ in st_1^m, the number of possible values is $2^{16.09+3+14} = 2^{33.09}$. The number of possible differences for rows $4, 6, 7, 8$ on the plaintext side is $2^{4\times6-0.57} = 2^{23.43}$. Consequently, the average number of candidate value pairs per difference for these rows in st_0^m is $2^{33.09-23.43} = 2^{9.66}$. Note that 6 bits in rows 7–11 of st_1^m have been determined in previous steps; additionally, 6 key bits recovered in this step overlap with those recovered in previous steps. Therefore, after this step, the average number of remaining tuples is $2^{156.69+9.66-2\times6-6} = 2^{148.35}$, and a total of 151 key bits are determined.

The time overhead in this step is dominated by hash-table lookups and XOR operations. The number of message pairs entering this phase is $2^{170.69}$. Since each message pair is subjected to one hash-table lookup in each of the six sub-steps, the time complexity of hash-table lookups is bounded by $2^{170.69} \times 6 \times 2^{-8.81} = 2^{164.46}$ encryptions. The entire step involves approximately $2^{179.12}$ XOR operations on full 192-bit states, which is equivalent to $2^{179.12} \times 2^{-11.29} = 2^{167.83}$ encryptions.[5] Therefore, the overall time complexity of this step is approximately $2^{164.46} + 2^{167.83} \approx 2^{167.96}$ encryptions. In addition, this step requires pre-storing all possible values of the active rows in st_0^m and st_3^c, and the total memory space is approximately $2^{61.4}$ complete 192-bit states.

At this point, for the remaining $2^{148.35}$ tuples, we have recovered all key bits corresponding to the active rows in k_0 on the plaintext side and k_7 on the ciphertext side. Consequently, we have determined the values of all active rows in st_1^m and st_2^c. In addition, the values of certain bits in some inactive rows of st_1^m and st_2^c are also determined, so as to ensure the differential propagation of the active rows through SB_2 and SB_{13}.

Step 3. Recover the key bits corresponding to the inactive rows on the plaintext side and the ciphertext side. First, recover the unknown key bits in the inactive rows of k_7 on the ciphertext side. Table 10 shows the number of determined state bits in the inactive rows

[5] On our experimental platform, one full 7-round encryption costs approximately $2^{11.29}$ times the time of one full 192-bit XOR operation.

of st_2^c and the number of determined key bits in the inactive rows of k_7 on the ciphertext side. This process is straightforward: handle each inactive row one by one, iterate over the unknown key bits, and check whether the constraint of the values of the inactive rows in st_2^c can be satisfied after SB_{14}^{-1}. This process requires $2^{148.35} \times 110$ S-box lookups, leaving an average of $2^{154.35}$ tuples, and a total of 176 key bits are determined. Next, recover the unknown key bits in the inactive rows of k_0 on the plaintext side. Table 11 shows the number of determined state bits in the inactive rows of st_1^m and the number of determined key bits in the inactive rows of k_0 on the plaintext side. The processing method is the same as that for the ciphertext side. This process requires $2^{154.35} \times 26$ S-box lookups, leaving an average of $2^{155.35}$ tuples, and a total of 189 key bits are determined. The total time complexity of this step is approximately $2^{150.34} \times 448$ S-box lookups, which does not exceed $2^{150.34}$ full 7-round encryptions.

Table 10: The number (num_v) of determined state bits in the inactive rows of st_2^c and the number (num_k) of determined key bits of k_7 in the inactive rows on the ciphertext side.

Row	0	1	6	9	10	13	14	16	17	18	25	27	28
num_v	2	2	3	1	1	1	1	1	1	1	1	2	2
num_k	3	3	6	4	2	6	1	6	4	5	6	6	1

Table 11: The number (num_v) of determined state bits in the inactive rows of st_1^m and the number (num_k) of determined key bits of k_0 in the inactive rows on the plaintext side.

Row	2	3	5	18	19	22	30	31
num_v	1	2	2	1	2	2	1	1
num_k	4	5	4	4	4	5	5	4

Step 4. Brute-forcing the remaining 3 key bits. The time complexity of this step is $2^{155.35+3} = 2^{158.35}$ encryptions.

The overall complexity analysis of the above key recovery attack on SPEEDY-7-192 are shown in Table 12.

Table 12: Overall complexity analysis.

	Data gen. phase	Step 1	Step 2	Step 3	Step 4	Total
Data complexity	$2^{173.55}$	–	–	–	–	$2^{173.55}$
Time complexity	$2^{173.55}$	$2^{170.66}$	$2^{167.96}$	$2^{150.34}$	$2^{158.35}$	$2^{173.76}$
Memory complexity	2^{66}	–	$2^{61.4}$	–	–	$2^{66.06}$

Success probability. The data complexity above is chosen so that the expected number of right pairs following the entire differential propagation path is about one. Under the standard Poisson approximation, the probability that at least one right pair is generated is therefore about $1 - e^{-1} \approx 63\%$. Conditioned on the existence of such a right pair, the correct key material is not discarded in Steps 1–3, and the remaining unknown key bits are exhaustively tested in Step 4. Hence, the conditional success probability of the key recovery phase is one, while the overall success probability of the attack is about 63%. This overall success probability can be amplified to $1 - e^{-\lambda}$ by increasing the amount of generated data so that the expected number of right pairs is λ.

5 Conclusion

In this paper, we presented a differential cryptanalysis of the SPEEDY family from two perspectives: security evaluation and key recovery attack. We proposed an efficient search method for differential characteristics based on *search patterns*. Through a careful decomposition of differential characteristics into two types of subtrails, *core trails* and *extended trails*, the method yields an efficient search framework. The proposed *short-round reuse via hash-map concatenation* technique further enables efficient search for core trails.

Unlike previous evaluations, our method imposes no a priori restrictions on the allowed differential characteristics. Computational pruning keeps the search tractable, while conservative upper bounds cover the pruned region, thereby enabling an accurate evaluation over the global space. We determined the exact minimum numbers of active S-boxes for 2-, 3-, and 5-round characteristics to be 13, 23, and 45, respectively, and narrowed the 4-round minimum from the previously reported 35 to $[33, 34]$. In particular, we found a 4-round characteristic with 34 active S-boxes outside the search spaces considered in previous evaluations. Moreover, we obtained tighter upper bounds on differential probability for 2 to 5 rounds than those reported previously. To the best of our knowledge, our results are the first to provide security bounds for SPEEDY against differential cryptanalysis over the global space, thereby more accurately reflecting its security strength. In addition, we obtained the highest-probability differential characteristics known to date for 2 to 5 rounds; the 2-round characteristic is globally optimal, and the 2-, 3-, and 4-round results disprove the designers' branch-number-based security claims.

Applying the quasidifferential framework to the searched differential characteristics, we identified several 5-round distinguishers with higher expected differential probabilities than those used in the existing key recovery attacks. Finally, by optimizing the probabilistic extension parameters used in Boura et al.'s key recovery attack [BDG+25b], we obtained the best known key recovery attack on SPEEDY-7-192, with data complexity $2^{173.48}$ and time complexity $2^{173.69}$ encryptions.

Overall, our work highlights the importance of evaluating security over the global space rather than relying on structurally restricted models. Extending the evaluation of SPEEDY against linear cryptanalysis is an interesting direction for future work.

References

[AK18] Ralph Ankele and Stefan Kölbl. Mind the gap - a closer look at the security of block ciphers against differential cryptanalysis. Cryptology ePrint Archive, Paper 2018/689, 2018. URL: https://eprint.iacr.org/2018/689.

[AKM+24] Zahra Ahmadian, Akram Khalesi, Dounia M'foukh, Hossein Moghimi, and María Naya-Plasencia. Improved differential meet-in-the-middle cryptanalysis. In Marc Joye and Gregor Leander, editors, *Advances in Cryptology - EUROCRYPT 2024 - 43rd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Zurich, Switzerland, May 26-30, 2024, Proceedings, Part I*, volume 14651 of *Lecture Notes in Computer Science*, pages 280–309. Springer, 2024. doi:10.1007/978-3-031-58716-0_10.

[BDBN23] Christina Boura, Nicolas David, Rachelle Heim Boissier, and María Naya-Plasencia. Better steady than speedy: Full break of SPEEDY-7-192. In Carmit Hazay and Martijn Stam, editors, *Advances in Cryptology - EUROCRYPT 2023 - 42nd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Lyon, France, April 23-27, 2023, Proceedings, Part IV*, volume 14007 of *Lecture Notes in Computer Science*, pages 36–66. Springer, 2023. doi:10.1007/978-3-031-30634-1_2.

[BDD+24] Christina Boura, Nicolas David, Patrick Derbez, Rachelle Heim Boissier, and María Naya-Plasencia. A generic algorithm for efficient key recovery in differential attacks - and its associated tool. In Marc Joye and Gregor Leander, editors, *Advances in Cryptology - EUROCRYPT 2024 - 43rd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Zurich, Switzerland, May 26-30, 2024, Proceedings, Part I*, volume 14651 of *Lecture Notes in Computer Science*, pages 217–248. Springer, 2024. doi:10.1007/978-3-031-58716-0_8.

[BDG25a] Christina Boura, Patrick Derbez, and Baptiste Germon. Extending the quasidifferential framework: From fixed-key to expected differential probability. *IACR Trans. Symmetric Cryptol.*, 2025(1):515–541, 2025. URL: https://doi.org/10.46586/tosc.v2025.i1.515-541, doi:10.46586/TOSC.V2025.I1.515-541.

[BDG+25b] Christina Boura, Patrick Derbez, Baptiste Germon, Rachelle Heim Boissier, and María Naya-Plasencia. SPEEDY: caught at last. In Goichiro Hanaoka and Bo-Yin Yang, editors, *Advances in Cryptology - ASIACRYPT 2025 - 31st International Conference on the Theory and Application of Cryptology and Information Security, Melbourne, VIC, Australia, December 8-12, 2025, Proceedings, Part I*, volume 16245 of *Lecture Notes in Computer Science*, pages 189–220. Springer, 2025. doi:10.1007/978-981-95-5018-0_7.

[BN24] Tim Beyne and Addie Neyt. Note on the cryptanalysis of speedy. Cryptology ePrint Archive, Report 2024/262, 2024. URL: https://eprint.iacr.org/2024/262.

[BN25] Tim Beyne and Addie Neyt. Improved differential cryptanalysis of SPEEDY. *IACR Trans. Symmetric Cryptol.*, 2025(3):444–474, 2025. URL: https://doi.org/10.46586/tosc.v2025.i3.444-474, doi:10.46586/TOSC.V2025.I3.444-474.

[BR22] Tim Beyne and Vincent Rijmen. Differential cryptanalysis in the fixed-key model. In Yevgeniy Dodis and Thomas Shrimpton, editors, *Advances in Cryptology - CRYPTO 2022 - 42nd Annual International Cryptology Conference, CRYPTO 2022, Santa Barbara, CA, USA, August 15-18, 2022, Proceedings, Part III*, volume 13509 of *Lecture Notes in Computer Science*, pages 687–716. Springer, 2022. doi:10.1007/978-3-031-15982-4_23.

[BS90] Eli Biham and Adi Shamir. Differential cryptanalysis of des-like cryptosystems. In Alfred Menezes and Scott A. Vanstone, editors, *Advances in Cryptology - CRYPTO '90, 10th Annual International Cryptology Conference, Santa Barbara, California, USA, August 11-15, 1990, Proceedings*, volume 537 of *Lecture Notes in Computer Science*, pages 2–21. Springer, 1990. doi:10.1007/3-540-38424-3_1.

[Hey20] Howard M. Heys. Key dependency of differentials: Experiments in the differential cryptanalysis of block ciphers using small s-boxes. Cryptology ePrint Archive, Paper 2020/1349, 2020. URL: https://eprint.iacr.org/2020/1349.

[Knu92] Lars R. Knudsen. Iterative characteristics of DES and s^2-des. In Ernest F. Brickell, editor, *Advances in Cryptology - CRYPTO '92, 12th Annual International Cryptology Conference, Santa Barbara, California, USA, August 16-20, 1992, Proceedings*, volume 740 of *Lecture Notes in Computer Science*, pages 497–511. Springer, 1992. doi:10.1007/3-540-48071-4_35.

[LMM91] Xuejia Lai, James L. Massey, and Sean Murphy. Markov ciphers and differential cryptanalysis. In Donald W. Davies, editor, *Advances in Cryptology - EUROCRYPT '91, Workshop on the Theory and Application of of Cryptographic Techniques, Brighton, UK, April 8-11, 1991, Proceedings*, volume 547 of *Lecture Notes in Computer Science*, pages 17–38. Springer, 1991. doi:10.1007/3-540-46416-6_2.

[LMMR21] Gregor Leander, Thorben Moos, Amir Moradi, and Shahram Rasoolzadeh. The SPEEDY family of block ciphers engineering an ultra low-latency cipher from gate level for secure processor architectures. *tches*, 2021(4):510–545, 2021. URL: https://tches.iacr.org/index.php/TCHES/article/view/9074, doi:10.46586/tches.v2021.i4.510-545.

[OCP] OCP: Open Cryptanalysis Platform. URL: https://github.com/Open-CP/OCP.

[OMA95] Kazuo Ohta, Shiho Moriai, and Kazumaro Aoki. Improving the search algorithm for the best linear expression. In Don Coppersmith, editor, *Advances in Cryptology - CRYPTO '95, 15th Annual International Cryptology Conference, Santa Barbara, California, USA, August 27-31, 1995, Proceedings*, volume 963 of *Lecture Notes in Computer Science*, pages 157–170. Springer, 1995. doi:10.1007/3-540-44750-4_13.

[SYC+24] Ling Song, Qianqian Yang, Yincen Chen, Lei Hu, and Jian Weng. Probabilistic extensions: A one-step framework for finding rectangle attacks and beyond. In Marc Joye and Gregor Leander, editors, *Advances in Cryptology - EUROCRYPT 2024 - 43rd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Zurich, Switzerland, May 26-30, 2024, Proceedings, Part I*, volume 14651 of *Lecture Notes in Computer Science*, pages 339–367. Springer, 2024. doi:10.1007/978-3-031-58716-0_12.

[WNL+23] Jinliang Wang, Chao Niu, Qun Liu, Muzhou Li, Bart Preneel, and Meiqin Wang. Cryptanalysis of SPEEDY. In Leonie Simpson and Mir Ali Rezazadeh Baee, editors, *Information Security and Privacy - 28th Australasian Conference, ACISP 2023, Brisbane, QLD, Australia, July 5-7, 2023, Proceedings*, volume 13915 of *Lecture Notes in Computer Science*, pages 124–156. Springer, 2023. doi:10.1007/978-3-031-35486-1_7.

[Zha24] Lei Zhang. Observations on the branch number and differential analysis of SPEEDY. *Des. Codes Cryptogr.*, 92(5):1175–1199, 2024. URL: https://doi.org/10.1007/s10623-023-01336-z, doi:10.1007/S10623-023-01336-Z.

[ZZC24] Chunning Zhou, Wentao Zhang, and Weiwei Cao. A new approach of evaluating the security against differential and linear cryptanalysis and its applications to serpent, NOEKEON and ASCON. *Comput. J.*, 67(1):274–291, 2024. URL: https://doi.org/10.1093/comjnl/bxac173, doi:10.1093/COMJNL/BXAC173.

A SPEEDY components

A.1 The SPEEDY S-box

Table 13: Table representation of the SPEEDY S-box.

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

A.2 The bit-permutation PB used in the key schedule of SPEEDY-r-192

Table 14: The bit-permutation PB used in the key schedule of SPEEDY-r-192.

i	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
$P(i)$	1	8	15	22	29	36	43	50	57	64	71	78	85	92	99	106
i	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
$P(i)$	113	120	127	134	141	148	155	162	169	176	183	190	5	12	19	26
i	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47
$P(i)$	33	40	47	54	61	68	75	82	89	96	103	110	117	124	131	138
i	48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63
$P(i)$	145	152	159	166	173	180	187	2	9	16	23	30	37	44	51	58
i	64	65	66	67	68	69	70	71	72	73	74	75	76	77	78	79
$P(i)$	65	72	79	86	93	100	107	114	121	128	135	142	149	156	163	170
i	80	81	82	83	84	85	86	87	88	89	90	91	92	93	94	95
$P(i)$	177	184	191	6	13	20	27	34	41	48	55	62	69	76	83	90
i	96	97	98	99	100	101	102	103	104	105	106	107	108	109	110	111
$P(i)$	97	104	111	118	125	132	139	146	153	160	167	174	181	188	3	10
i	112	113	114	115	116	117	118	119	120	121	122	123	124	125	126	127
$P(i)$	17	24	31	38	45	52	59	66	73	80	87	94	101	108	115	122
i	128	129	130	131	132	133	134	135	136	137	138	139	140	141	142	143
$P(i)$	129	136	143	150	157	164	171	178	185	0	7	14	21	28	35	42
i	144	145	146	147	148	149	150	151	152	153	154	155	156	157	158	159
$P(i)$	49	56	63	70	77	84	91	98	105	112	119	126	133	140	147	154
i	160	161	162	163	164	165	166	167	168	169	170	171	172	173	174	175
$P(i)$	161	168	175	182	189	4	11	18	25	32	39	46	53	60	67	74
i	176	177	178	179	180	181	182	183	184	185	186	187	188	189	190	191
$P(i)$	81	88	95	102	109	116	123	130	137	144	151	158	165	172	179	186

B Experimental setup

All experiments were conducted on a desktop computer equipped with a 13th Gen Intel® Core(TM) i7-13700 CPU, 32.0 GB of RAM, and Windows 11 Pro for Workstations (version 25H2, OS build 26200.8246). We used Gurobi Interactive Shell (win64), version 12.0.3, as

the MILP solver. Unless otherwise stated, Gurobi's default thread setting was used. For the parallel search procedures, we used 14 worker processes.

C Summary of the differential characteristics found in our search

Here, we present the differential characteristics with the minimum number of active S-boxes, the best differential probability, and the best expected differential probability in Appendices C.1, C.2, and C.3, respectively. Note that the term "best" here refers to the best values found within our search, not the global maximum over the entire search space. Each round of the differential characteristic involves four differential states, denoted as $\mathrm{st}_0^i, \mathrm{st}_1^i, \mathrm{st}_2^i, \mathrm{st}_3^i$ (i denotes the round number), which correspond to the states before and after the two SB layers, and each state is represented in hexadecimal notation. Concretely, each hexadecimal string is first converted into a binary string of length 192 bits. Then, following row-major order (from left to right and top to bottom), the binary bits can be arranged to fill a 32×6 two-dimensional state.

C.1 Differential characteristics with the minimum number of active S-boxes

The minimum numbers of active S-boxes in the differential characteristics for 2, 3, and 5 rounds are 13, 23, and 45, respectively; for 4 rounds, the minimum lies in the interval $[33, 34]$, and the smallest value we actually found for 4 rounds is 34. In Appendix C.2, the 2-round differential characteristic shown in Table 17 has 13 active S-boxes, and the 5-round differential characteristic shown in Table 20 has 45 active S-boxes. Therefore, we only present here the 3-round differential characteristic with 23 active S-boxes and the 4-round differential characteristic with 34 active S-boxes, as shown in Tables 15 and 16, respectively.

Table 15: 3-round characteristic with 23 active S-boxes.

Round	State	Trail value (hex)
Round 1	st_0^0	0x0000000280000000000000000000000000000000000000000000
	st_1^0	0x0000000010000000000000000000000000000000000000000000
	st_2^0	0x0400000000000000000000000000000000000000000000000000
	st_3^0	0x4000000000000000000000000000000000000000000000000000
Round 2	st_0^1	0x0000000100000004000000040000000040000400000410
	st_1^1	0x0000000040000001000000010000000100000100000104
	st_2^1	0x0001000000040000000400000000400000400000 4100000
	st_3^1	0x0008000000200000002000000020000020000020800000
Round 3	st_0^2	0x0000000800800000000000000000000800800800000000
	st_1^2	0x0000008002000000000000000000000800200080000000
	st_2^2	0x000000a000000000000000000000000a800000000000000
	st_3^2	0x0000000400000000000000000000000080000000000000
#AS = 23		

Table 16: 4-round characteristic with 34 active S-boxes.

Round	State	Trail value (hex)
Round 1	st_0^0	0x0004000000000000000000000000000000000000000000000000
	st_1^0	0x0001000000000000000000000000000000000000000000000000
	st_2^0	0x0000000000000000000000000000000000000000000000000004
	st_3^0	0x0000000000000000000000000000000000000000000000000002
Round 2	st_0^1	0x0020000000800000000800000000800000800000082000000
	st_1^1	0x0100000040000000004000000240000240000060000000
	st_2^1	0x4000000100010000000010002010002010000000200000000
	st_3^1	0x2000000080080000000800020800020800000000800000000
Round 3	st_0^2	0x0000000000000000000000000000820000000000000200
	st_1^2	0x0000000000000000000000000002040000000000000040
	st_2^2	0x0000000000000000000000000003000000000100000000
	st_3^2	0x0000000000000000000000000001000000001000000000
Round 4	st_0^3	0x0000000000000010010010410000010000000001040000000
	st_1^3	0x0000000000000100040018100000010000000010200000000
	st_2^3	0x000000000000540000000c400000000000006000000000000
	st_3^3	0x0000000000008c0000000e400000000000030000000000000
#AS = 34		

C.2 Differential characteristics with the best differential probability

The best differential probabilities of the differential characteristics we searched for 2, 3, 4, and 5 rounds are $2^{-46.245}$, $2^{-76.15}$, $2^{-119.735}$, and $2^{-156.565}$, respectively, as shown in Tables 17, 18, 19, and 20. For 2 rounds, the result is the maximum differential probability; for 3, 4, and 5 rounds, the results are not the maximum in the global search space.

Table 17: 2-round characteristic with differential probability $2^{-46.245}$.

Round	State	Trail value (hex)
Round 1	st_0^0	0x0000000280000000000000000000000000000000000000000000
	st_1^0	0x0000000010000000000000000000000000000000000000000000
	st_2^0	0x0400000000000000000000000000000000000000000000000000
	st_3^0	0x0400000000000000000000000000000000000000000000000000
Round 2	st_0^1	0x0010000000400000000400000000400000400000041000000
	st_1^1	0x0200000000400000004000000004000000400000810000000
	st_2^1	0x0210000000000000100000000110000000000000c00000000
	st_3^1	0x03c0000000000000800000001100000000000000400000000
Differential Probability = $2^{-46.245}$		

Table 18: 3-round characteristic with differential probability $2^{-76.15}$.

Round	State	Trail value (hex)
Round 1	st_0^0	0x0000000280000000000000000000000000000000000000000
	st_1^0	0x0000000010000000000000000000000000000000000000000
	st_2^0	0x0400000000000000000000000000000000000000000000000
	st_3^0	0x4000000000000000000000000000000000000000000000000
Round 2	st_0^1	0x0000000100000040000000040000000040000400000410
	st_1^1	0x0000000040000001000000001000000010000010000104
	st_2^1	0x0001000000040000000400000000400000400000410000
	st_3^1	0x0004000000100000001000000010000010000010400000
Round 3	st_0^2	0x0000000000100100000000000000000000000010010010000
	st_1^2	0x0000000000200080000000000000000000000008020008000
	st_2^2	0x0000000000280000000000000000000000008000028000000
	st_3^2	0x0000000000010000000000000000000000000500001b000000
Differential Probability = $2^{-76.15}$		

Table 19: 4-round characteristic with differential probability $2^{-119.735}$.

Round	State	Trail value (hex)
Round 1	st_0^0	0x00000000000000000000000000000000000000000000000001
	st_1^0	0x00000000000000000000000000000000000000000000000004
	st_2^0	0x00000000000000000000000000000000000000000000100000
	st_3^0	0x00000000000000000000000000000000000000000000080000
Round 2	st_0^1	0x00002000000002000000002000002000002080000000080
	st_1^1	0x00004000000004000000004000004000004100000000100
	st_2^1	0x10000000010000000100000100000104000000004000000
	st_3^1	0x08000000080000000800000800000820000000002000000
Round 3	st_0^2	0x00000000000000000000002002002000000000000002002
	st_1^2	0x00000000000000000000002008002000000000000004001
	st_2^2	0x0000000000000000000002a000000000000000140000000
	st_3^2	0x00000000000000000000002000000000000000080000000
Round 4	st_0^3	0x00208000008200008200200000000002000000000000000
	st_1^3	0x00810000040800040800200000000008000000000000000
	st_2^3	0x00000001800001a000000000000080000000000000000c
	st_3^3	0x00000000c00001e000000000000050000000000000000001
Differential Probability = $2^{-119.735}$		

Table 20: 5-round characteristic with differential probability $2^{-156.565}$.

Round	State	Trail value (hex)
Round 1	st_0^0	0x0000000000000000000000a0000000000000000000000000
	st_1^0	0x0000000000000000000000040000000000000000000000000
	st_2^0	0x0000000000000000100000000000000000000000000000000
	st_3^0	0x0000000000000000100000000000000000000000000000000
Round 2	st_0^1	0x0010000010400000000400000010000000010000000001000
	st_1^1	0x0100000104000000004000000100000000100000000010000
	st_2^1	0x4000004100000000100000004000000040000000400000
	st_3^1	0x2000002080000000080000002000000020000000200000
Round 3	st_0^2	0x0080080080000000000000800800000000000000000000
	st_1^2	0x0080200080000000000001000400000000000000000000
	st_2^2	0x0000280000000000000050000000000000000000000008
	st_3^2	0x0000020000000000000008000000000000000000000002
Round 4	st_0^3	0x0000000008000200008000000080002000000000000082
	st_1^3	0x0000000088002200020000000200020000000000000084
	st_2^3	0x0000800028000202000000020000002000000080100000
	st_3^3	0x0001000041000041000000010000000400000100100000
Round 5	st_0^4	0x0000000000041000000000000010000000000000000000
	st_1^4	0x0000000000040800000000000040000000000000000000
	st_2^4	0x0000001800000000000000000100000000000000000000
	st_3^4	0x0000004c000000000000000020000000000000000000
Differential Probability = $2^{-156.565}$		

C.3 Differential characteristics with the best expected differential probability

The best expected differential probabilities of the differential characteristics we searched for 2, 3, 4, and 5 rounds are $2^{-46.245}$, $2^{-74.998}$, $2^{-117.32}$, and $2^{-152.565}$, respectively. In Appendix C.2, the expected differential probability of the 2-round differential characteristic shown in Table 17 is $2^{-46.245}$, and the expected differential probability of the 3-round differential characteristic shown in Table 18 is $2^{-74.998}$. Therefore, we only present here the 4-round differential characteristic with expected differential probability $2^{-117.32}$ and the 5-round differential characteristic with expected differential probability $2^{-152.565}$, as shown in Tables 21 and 22, respectively.

Table 21: 4-round characteristic with expected differential probability $2^{-117.32}$.

Round	State	Trail value (hex)
Round 1	st_0^0	0x00000000000000000000000000000000000000000000000001
	st_1^0	0x00000000000000000000000000000000000000000000000004
	st_2^0	0x00000000000000000000000000000000000000000000100000
	st_3^0	0x00000000000000000000000000000000000000000080000000
Round 2	st_0^1	0x00002000000002000000002000002000002080000000080
	st_1^1	0x00004000000004000000004000004000004100000000100
	st_2^1	0x10000000010000000010000010000010400000000004000000
	st_3^1	0x08000000008000000008000008000008200000000002000000
Round 3	st_0^2	0x00000000000000000000002002002000000000000002002
	st_1^2	0x00000000000000000000002008002000000000000004001
	st_2^2	0x0000000000000000000002a0000000000000140000000
	st_3^2	0x00000000000000000000002000000000000000080000000
Round 4	st_0^3	0x00208000008200008200200000000002000000000000000
	st_1^3	0x00810000040800040800200000000008000000000000000
	st_2^3	0x00000001800001a000000000000080000000000000000c
	st_3^3	0x00000000c00001e0000000000000500000000000000038
EDP = $2^{-117.32}$		

Table 22: 5-round characteristic with expected differential probability $2^{-152.565}$.

Round	State	Trail value (hex)
Round 1	st_0^0	0x0000000000000000000000a0000000000000000000000000
	st_1^0	0x0000000000000000000000040000000000000000000000
	st_2^0	0x0000000000000000100000000000000000000000000000
	st_3^0	0x0000000000000000100000000000000000000000000000
Round 2	st_0^1	0x0010000010400000000400000010000000010000000001000
	st_1^1	0x0100000104000000004000000100000000100000000010000
	st_2^1	0x4000004100000000100000004000000040000000400000
	st_3^1	0x2000002080000000080000002000000020000000200000
Round 3	st_0^2	0x0080080080000000000000800800000000000000000000
	st_1^2	0x0080200080000000000001000400000000000000000000
	st_2^2	0x0000280000000000000050000000000000000000000008
	st_3^2	0x0000020000000000000008000000000000000000000002
Round 4	st_0^3	0x0000000008000200008000000080002000000000000082
	st_1^3	0x0000000088002200020000000200020000000000000084
	st_2^3	0x0000800028000202000000020000002000000080100000
	st_3^3	0x0001000041000041000000010000004000000100100000
Round 5	st_0^4	0x0000000000041000000000000010000000000000000000
	st_1^4	0x0000000000040800000000000040000000000000000000
	st_2^4	0x0000001800000000000000000100000000000000000000
	st_3^4	0x0000001800000000000000002100000000000000000000
EDP = $2^{-152.565}$		

D Dominant differential trails of the distinguisher used for key recovery attack on SPEEDY-7-192

Table 23 and Table 24 are the two dominant 5-round differential trails of the distinguisher used for the key recovery attack on SPEEDY-7-192, both with expected differential probability $2^{-159.565}$. In addition, there are 13 other minor 5-round differential trails. Clustering these 15 trails yields the attack distinguisher with an expected differential probability of $2^{-158.36}$.

Table 23: One of the two dominant 5-round differential trails of the distinguisher used for the attack on SPEEDY-7-192 of expected differential probability $2^{-159.565}$.

Round	State	Trail value (hex)
Round 1	st_0^0	0x00000000000000000000000104000000000000000000000
	st_1^0	0x00000000000000000000000810000000000000000000000
	st_2^0	0x0000000000000000000000c000000000000000000000000
	st_3^0	0x00000000000000000000000400000000000000000000000
Round 2	st_0^1	0x00004000004000004100000001000000040000000040000
	st_1^1	0x00040000040000041000000010000000400000000400000
	st_2^1	0x01000001000001040000000040000001000000010000000
	st_3^1	0x00800000800000820000000200000008000000008000000
Round 3	st_0^2	0x00000000200200200000000000000200200000000000000
	st_1^2	0x00000000200800200000000000000400100000000000000
	st_2^2	0x000000200000a000000000000000140000000000000000
	st_3^2	0x00000080000080000000000000002000000000000000000
Round 4	st_0^3	0x00002080000000002000800020000000020000800000000
	st_1^3	0x00002100000000022000880008000000008000800000000
	st_2^3	0x0040000000020000a000080800000000800000080000000 2
	st_3^3	0x0040000000040001040001040000000040000001000000 4
Round 5	st_0^4	0x00000000000000000104000000000000004000000000000
	st_1^4	0x00000000000000000102000000000000004000000000000
	st_2^4	0x00000000000000600000000000000010000000000000000
	st_3^4	0x00000000000001000000000000000004000000000000000
EDP = $2^{-159.565}$		

Table 24: The other dominant 5-round differential trail of the distinguisher used for the attack on SPEEDY-7-192 of expected differential probability $2^{-159.565}$.

Round	State	Trail value (hex)
Round 1	st_0^0	0x000000000000000000000000104000000000000000000000
	st_1^0	0x000000000000000000000000810000000000000000000000
	st_2^0	0x000000000000000000000000c000000000000000000000
	st_3^0	0x000000000000000000000000400000000000000000000000
Round 2	st_0^1	0x000040000040000041000000001000000040000000040000
	st_1^1	0x000400000400000410000000010000000400000000400000
	st_2^1	0x010000010000010400000000400000010000000010000000
	st_3^1	0x008000008000008200000000200000008000000008000000
Round 3	st_0^2	0x000000000200200200000000000000200200000000000000
	st_1^2	0x000000000200800200000000000000400100000000000000
	st_2^2	0x000000200000a000000000000000140000000000000000
	st_3^2	0x000000080000080000000000000002000000000000000000
Round 4	st_0^3	0x000002080000000002000800020000000020000800000000
	st_1^3	0x000008040000000022000880008000000008000800000000
	st_2^3	0x008000000020000a000080800000000800000800000001
	st_3^3	0x004000000040001040001040000000040000001000000004
Round 5	st_0^4	0x000000000000000000104000000000000004000000000000
	st_1^4	0x000000000000000000102000000000000004000000000000
	st_2^4	0x000000000000006000000000000001000000000000000000
	st_3^4	0x000000000000010000000000000004000000000000000000
EDP = $2^{-159.565}$		

E Key bits of the round key k_7

Table 25: Key bits of the round key k_7

Row	Master key bits					
0	161	168	175	182	189	4
1	11	18	25	32	39	46
2	53	60	67	74	81	88
3	95	102	109	116	123	130
4	137	144	151	158	165	172
5	179	186	1	8	15	22
6	29	36	43	50	57	64
7	71	78	85	92	99	106
8	113	120	127	134	141	148
9	155	162	169	176	183	190
10	5	12	19	26	33	40
11	47	54	61	68	75	82
14	173	180	187	2	9	16
15	23	30	37	44	51	58
16	65	72	79	86	93	100
17	107	114	121	128	135	142
18	149	156	163	170	177	184
19	191	6	13	20	27	34
26	101	108	115	122	129	136
27	143	150	157	164	171	178
28	185	0	7	14	21	28
29	35	42	49	56	63	70
30	77	84	91	98	105	112
31	119	126	133	140	147	154

F Data used in candidate 1 attack

Figure 6 is the schematic diagram of the candidate 1 attack on SPEEDY-7-192, where $(l_{\text{in}}, l_{\text{out}}) = (11, 20)$, and $p' = p_{\text{in}} + p + p_{\text{out}} = 6.17 + 158.36 + 7.95 = 172.48$. Table 26 presents the filter values of active rows in SB_1 and SB_{14}, and Table 27 presents the number of possible values for the active rows in st_2^m and st_1^c; they are used in step 1 and step 2 during the key recovery process, respectively. Using a similar key recovery procedure as in Section 4.2, the data complexity and time complexity of this attack are approximately $2^{173.48}$ and $2^{173.69}$, respectively, and its memory complexity is 2^{72} complete 192-bit states.

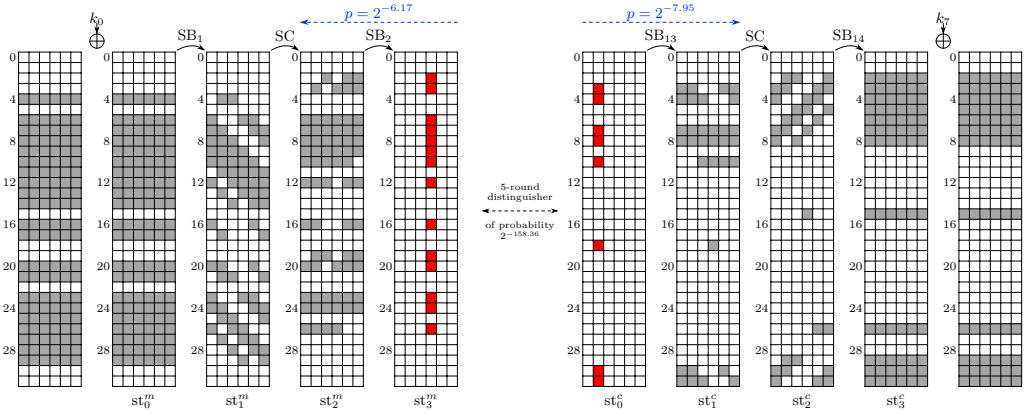

Figure 6: The schematic diagram of the candidate 1 attack on SPEEDY-7-192.

Table 26: Filter values (in $\log_2$) of active rows in SB_1 and SB_{14} of the candidate 1 attack.

SB_1				SB_{14}	
Row	Filter	Row	Filter	Row	Filter
4	0.476	20	0.117	2	0.046
6	0.069	21	0.476	3	0
7	0.023	23	0.046	4	0.023
8	0	24	0.023	5	0.046
9	0	25	0.476	6	0.167
10	0	26	0.069	7	0.069
11	0	27	0.117	8	0.752
12	0	28	0.167	15	1.541
13	0	29	0.642	26	0.956
14	0.167			29	0.752
16	0.385			30	0.046
17	0.476			31	0.093
Total filter: 8.22					

Table 27: The number (in $\log_2$) of possible values for the active rows in st_2^m and st_1^c of the candidate 1 attack, i.e., each entry m corresponds to 2^m possible values.

st_2^m				st_1^c	
Row	m	Row	m	Row	m
2	5.09	16	5.17	3	5
3	5	19	5.39	4	5.46
6	6	20	5.46	7	6
7	6	23	6	8	6
8	6	24	6	10	4.81
9	6	26	4.7	18	3
10	5.32			30	4.32
12	5.7			31	5.46