Hardware Performance Characterization of Block Cipher Structures[*]

Lu Xiao and Howard M. Heys
Electrical and Computer Engineering
Faculty of Engineering and Applied Science
Memorial University of Newfoundland
St. John's, NF, Canada A1B 3X5
{xiao, howard}@engr.mun.ca

Abstract

In this paper, we present a general framework for evaluating the performance characteristics of block cipher structures composed of S-boxes and Maximum Distance Separable (MDS) mappings. In particular, we examine nested Substitution-Permutation Networks (SPNs) and Feistel networks with round functions composed of S-boxes and MDS mappings. Within each cipher structure, many cases are considered based on two types of S-boxes (i.e., 4×4 and 8×8) and parameterized MDS mappings. In our study of each case, the hardware complexity and performance are analyzed. Cipher security, in the form of resistance to differential, linear, and Square attacks, is used to determine the minimum number of rounds required for a particular parameterized structure. Because the discussed structures are similar to many existing ciphers (e.g., Rijndael, Camellia, Hierocrypt, and Anubis), the analysis provides a meaningful mechanism for seeking efficient ciphers through a wide comparison of performance, complexity, and security.

1 Introduction

In product ciphers like DES [1] and Rijndael [2], the concepts of confusion and diffusion are vital to security. The Feistel network and the Substitution-Permutation Network (SPN) are two typical architectures to achieve this. In both architectures, Substitution-boxes (S-boxes) are typically used to perform substitution on small sub-blocks. An S-box is a nonlinear mapping from input bits to output bits, which meets many security requirements. In many recently proposed block ciphers (e.g., Rijndael, Hierocrypt [3], Anubis [4], and Khazad [5]),

[*]Presented at Cryptographers' Track - RSA Conference 2003, San Francisco, LNCS 2612, Springer-Verlag.

the outputs of a layer of parallel S-boxes are passed through a linear transformation based on a Maximum Distance Separable (MDS) code.

In this paper, the performance of several cipher structures is considered in terms of hardware time and space complexity. A performance comparison is made between different parameterized cases of 128-bit block ciphers in relation to security requirements. In the analysis, the hardware complexities of S-boxes and MDS mappings are based on the upper bounds of the minimum hardware complexity deduced in [6]. For a general invertible S-box, the upper bounds of the gate count and delay are obtained from the logic minimization of a hardware-efficient S-box model; for an MDS mapping, the upper bounds of the gate count and delay are obtained by searching MDS candidates for an optimal one when implemented by bit-parallel multipliers. Hence, the structures discussed in this paper are constructed with optimized components to produce high efficiencies in their categories. A conventional evaluation approach is taken in [6] with the space complexity evaluated by the number of bit-wise invertors and 2-input gates and the time complexity evaluated by the number of traversed layers in the gate network. In this paper, a weight is associated with different types of gates to distinguish their discrepancies in hardware cost. Performance metrics are defined for hardware with consideration of complexity and security.

Many ciphers are derived from appropriate configurations of S-boxes and linear transformations (typically MDS mappings). Rijndael, Hierocrypt, and Anubis can be regarded as specific cases of nested SPNs [3]. On the other hand, the round function of a Feistel network may contain one or several layers of S-boxes followed by a linear transformation such as an MDS mapping. For example, Camellia [7] is such a cipher with one layer of S-boxes in the round function (although the linear transformations are not MDS). In this paper, many cases of these cipher structures will be analyzed for their hardware complexities and performances.

2 Background

2.1 Properties of S-boxes

The properties of the S-boxes in a cipher are important in the consideration of a cipher's security against differential cryptanalysis [8] and linear cryptanalysis [9]. An $m \times n$ S-box, S, performs a mapping from an m-bit input X to an n-bit output Y. Considering all S-boxes, $\{S_i\}$, in a cipher, the maximum differential probability p_s is defined as:

$$p_s = \max_i \max_{\triangle X \neq 0, \triangle Y} prob\{S_i(X) \oplus S_i(X \oplus \triangle X) = \triangle Y\}$$

where "$\oplus$" denotes a bitwise XOR and "$\triangle$" denotes a bitwise XOR difference. The maximum linear probability is defined as:

$$q_s = \max_i \max_{\mathit{\Gamma} Y \neq 0, \mathit{\Gamma} X} (2 \times prob\{X \cdot \mathit{\Gamma} X = S_i(X) \cdot \mathit{\Gamma} Y\} - 1)^2$$

where "$\cdot$" denotes a bitwise inner product and $\mathit{\Gamma} X$ and $\mathit{\Gamma} Y$ denote masking variables. In this paper, all 4×4 S-boxes are assumed to satisfy $p_s, q_s \leq 2^{-2}$ and all 8×8 S-boxes are assumed to satisfy $p_s, q_s \leq 2^{-6}$. Many proposed ciphers such as Serpent [10], Rijndael, Hierocrypt-3 [11], and Camellia have S-boxes with these features; others such as Anubis and Khazad have slightly higher p_s and q_s.

2.2 MDS Mappings

A linear code over Galois field GF(2^n) is denoted as a (k, m, d)-code [12], where k is the symbol length of the encoded message, m is the symbol length of the original message, and d is the minimal symbol distance between any two encoded messages. An (k, m, d)-code is MDS if $d = k - m + 1$. In particular, a $(2m, m, m+1)$-code with generation matrix $\mathcal{G} = [\mathcal{I}|\mathcal{C}]$ where $\mathcal{C}$ is an $m \times m$ matrix and $\mathcal{I}$ is an identity matrix, determines an MDS mapping from the input $\mathcal{X}$ to the output $\mathcal{Y}$ through matrix multiplication over a Galois field as follows:

$$f_M : \mathcal{X} \mapsto \mathcal{Y} = \mathcal{C} \cdot \mathcal{X} \tag{1}$$

where

$$\mathcal{X} = \begin{pmatrix} X_0 \\ \vdots \\ X_{m-1} \end{pmatrix}, \quad \mathcal{Y} = \begin{pmatrix} Y_0 \\ \vdots \\ Y_{m-1} \end{pmatrix}, \quad \mathcal{C} = \begin{pmatrix} C_{0,0} & \dots & C_{0,m-1} \\ \vdots & \ddots & \vdots \\ C_{m-1,0} & \dots & C_{m-1,m-1} \end{pmatrix}.$$

Every entry in $\mathcal{X}, \mathcal{Y}$, and $\mathcal{C}$ is an element in GF(2^n).

When an invertible linear transformation $f : \mathcal{X} \mapsto \mathcal{Y}$ is used in a cipher, the avalanche effect which creates resistance to differential and linear attacks may be measured with its branch number B, which is defined as [13]:

$$B = \min_{\mathcal{X} \neq 0} \{H(\mathcal{X}) + H(\mathcal{Y})\}$$

where $H(\mathcal{X})$ and $H(\mathcal{Y})$ denotes the numbers of nonzero elements in $\mathcal{X}$ and $\mathcal{Y}$. It is proved that an MDS mapping as defined in (1) has an optimal branch number B equal to $m + 1$.

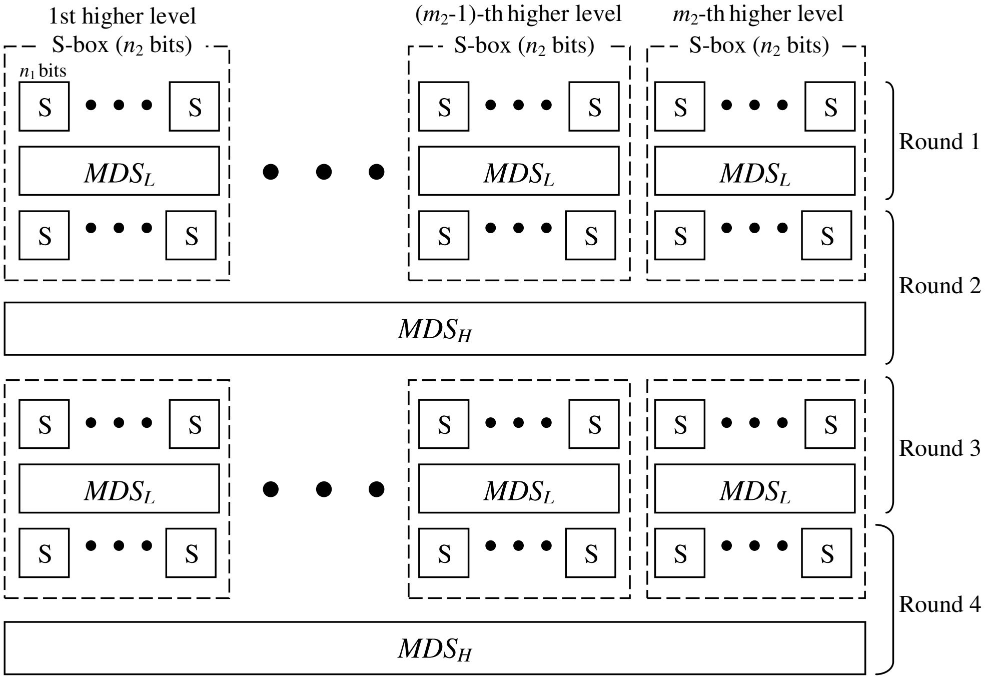
2.3 Nested SPNs

The concept of a nested SPN was first introduced in [3]. In a nested SPN, S-boxes may be viewed at different levels: each S-box at a higher level is actually a small SPN at the lower level. In this paper, we examine nested SPNs which have the following properties:

- The structure contains just two levels of SPNs. A higher level S-box consists of a lower level SPN; a lower level S-box is a real 4×4 or 8×8 S-box.
- The linear transformation layers in both levels are based on MDS codes, denoted as MDS_H for the higher level and MDS_L for the lower level.
- The round key mixture occurs directly before each layer of actual (i.e., lower-level) S-boxes. One additional subkey mixture is appended at the end of the cipher structure. The subkey bits are mixed with data bits by XOR operations.

- A "round" refers to the combination of the subkey mixture, lower-level S-box layer, and subsequent MDS_L or MDS_H linear transformation.

As Figure 1 shows, MDS_L is an MDS mapping from a $(2m_1, m_1, m_1+1)$-code over GF(2^{n_1}), while MDS_H is an MDS mapping from a $(2m_2, m_2, m_2+1)$-code over GF(2^{n_2}). The variables m_1, m_2, n_1, and n_2 represent parameter choices for a nested SPN.



MDS_L: based on a $(2m_1, m_1, m_1+1)$-code over GF(2^{n_1})

MDS_H: based on a $(2m_2, m_2, m_2+1)$-code over GF(2^{n_2})

Figure 1: Basic 2-level Nested SPN (4 Rounds)

In the most straightforward case, the output of each S-box forms one source symbol for the MDS mapping, and each encoded symbol forms the input of a subsequent S-box at the same level. So the size of an S-box is n_1 bits at the lower level and n_2 bits at the higher level. This leads to $n_2 = n_1 m_1$. Thus, the block size of the SPN is $n_1 m_1 m_2$. For example, Hierocrypt (Type I) is described as the iteration of such a 4-round structure where $n_1 = 8$, $n_2 = 32$, and $m_1 = m_2 = 4$.

At each level of a nested SPN, the branch number of the MDS layer determines the minimum number of active S-boxes in differential or linear cryptanalysis. For 4 rounds of a nested SPN, an active S-box at the higher level contains at least $m_1 + 1$ active S-boxes at the lower level. Since there are at least $m_2 + 1$ active S-boxes at the higher level, the minimum number of active lower-level S-boxes is $(m_1 + 1)(m_2 + 1)$. Therefore, the security against differential and linear attacks is evaluated as the following:

Theorem 1 (deduced from [2][3][13][14]): *With the assumption that all S-box approximations involved in linear and differential cryptanalysis are independent, for 4 rounds of a nested*

SPN the maximum differential characteristic probability is upper bounded by $p_s^{(m_1+1)(m_2+1)}$ *and the maximum linear characteristic probability is upper bounded by* $q_s^{(m_1+1)(m_2+1)}$.

The basic operations in MDS codes are multiplications and additions in finite fields. When n_2 is large, operations over GF(2^{n_2}) are inefficient and MDS_H can be costly in computation. An alternative method to obtain the same branch number is to concatenate several parallel MDS codes over a smaller finite field. The concatenated codes may be designed to ease a bitslice implementation.

Theorem 2 [3]: *An MDS mapping defined by a* $(2m, m, m+1)$*-code over the* nl*-bit symbol set can be constructed by concatenating* l *mappings defined by a* $(2m, m, m+1)$*-code over the* n*-bit symbol set, where* l *can be any positive integer.*

For the example illustrated by Figure 1, since $n_2 = m_1 n_1$, the mapping MDS_H over GF(2^{n_2}) can be implemented with m_1 parallel MDS mappings over GF(2^{n_1}). In this case, the basic MDS_H layer is denoted as $1 \times (2m_2, m_2, m_2+1)$ over GF(2^{n_2}), and its simplified MDS_H layer is denoted as $m_1 \times (2m_2, m_2, m_2+1)$ over GF(2^{n_2}) where n_2 is now the size of a smaller field and for this case $n_2 = n_1$. Since $m_1 n_1$ may be factored in other ways, other simplifications are also possible. Hence we can consider that the general relation $n_2 l = m_1 n_1$ can be used to determine different cases of MDS_H defined by the values of the symbol size, n_2, or the number of parallel MDS mappings, l. A similar approach can also be applied to the MDS_L layer. However, restrictions on values of n and m must be considered for designing a $(2m, m, m+1)$-code over GF(2^n) such that $2m \leq 2^n + 1$ [12].

The 128-bit ciphers Square, Rijndael, and Anubis can be regarded as the iterations of 4-round nested SPNs where $n_1 = n_2 = 8$, $m_1 = m_2 = 4$. The parameters of Hierocrypt (Type II) are selected as $n_1 = 8$, $n_2 = 4$, $m_1 = m_2 = 4$.

2.4 One Type of Feistel Networks

As a typical form of block ciphers, the Feistel network has been widely used and studied. In each round i of a Feistel network as shown in Figure 2(a), the right half of the round input (denoted as X_i) goes through an F-function parameterized by round key K_i. The output of the F-function (denoted as Y_i) is XORed with the left half of the round input. The round output is the swapped result of X_i and $X_{i-1} \oplus Y_i$. The F-function is also called the round function.

Figure 2(b) illustrates a subset of Feistel networks, which has a one round SPN inside F-function. The F-function includes one layer of key addition with K_i, one layer of invertible[1] S-boxes for substitution, and an MDS mapping layer as a linear transformation. If the MDS mapping layer is constructed through concatenation of several small MDS mappings, it is

[1] Invertible S-boxes are used so that a bijective round function can be constructed, which achieves the given upper bounds of maximal differential and linear probabilities faster in rounds than a general round function [15].

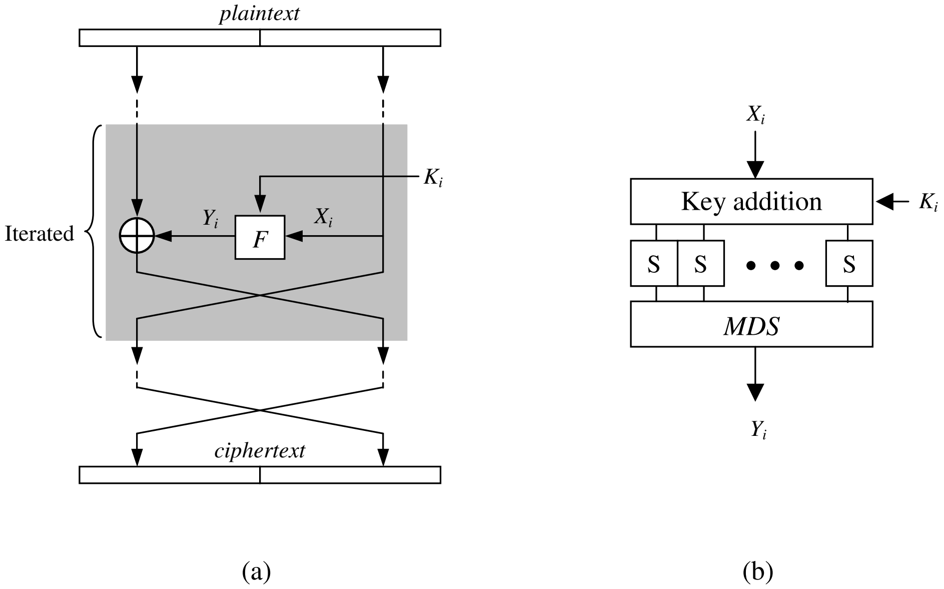


Figure 2: (a) General Encryption Dataflow of a Feistel Network (b) One Type of F-function

necessary to include a permutation of MDS symbols in the linear transformation in order to ensure the avalanche effect.

In a Feistel network whose round function has an invertible linear transformation appended to parallel S-boxes, it is proved in [16] that the number of active S-boxes in any differential or linear characteristic of $4r$ rounds is lower bounded by $r \times B + \lfloor r/2 \rfloor$, where B is the branch number of the linear transformation. Therefore, we get:

Theorem 3 (deduced from [16]): *In an $4r$ Feistel cipher with a round function as Figure 2(b) shows, the maximum differential and linear characteristic probabilities are upper bounded by $p_s^{r \times B + \lfloor r/2 \rfloor}$ and $q_s^{r \times B + \lfloor r/2 \rfloor}$, respectively.*

3 Comparison of Hardware Performance

It is normally hard to compare hardware performance among different block ciphers. The main problems are: 1) each implementation represents a tradeoff between area and delay; 2) the specific hardware cost of a gate network is dependent on the target technology; 3) ciphers may contain different security margins.

For the first problem, the classical delay-area product is used to evaluate the hardware complexity universally. The typical methods used in the hardware implementation of a block cipher include a round iterated design, a pipelined design, a loop-unrolled design, and a block parallel design [18]. For a given cipher, the delay-area product is kept roughly unchanged across the different design methods (except for a loop-unrolled design), assuming the control overhead for parallelism can be ignored. If a round iterated design is regarded as a reference, a k-block parallel design using several round iterated implementations will cost about k times the number of gates and result in about $1/k$ of the average time to produce an encrypted block. The same situation occurs in a pipelined design when each stage performs one or several rounds of the cipher. For loop unrolling, when k rounds are unrolled, the gate count will increase over an iterative design, but the average encryption time can be reduced. Loop unrolling usually results in low performance in the sense of the delay-area product.

For the second problem, a universal way is to assume that all gates have the same hardware cost [17]. Thus, the gate count and delay of all components are deduced from the upper bound of typical implementations. Such an approach leads to a measure of complexity which is technology-independent. However, in a certain target VLSI technology, the hardware costs of different gates may not be similar. In this case, it is possible to estimate the overall area (respectively, delay) by summing weighted gate counts (respectively, weighted gate layers traversed). The weights are proportional to the size of a gate (respectively, delay) and can be calculated by statistical comparison of hardware among gates based on a target technology. The hardware complexity is then evaluated by weighted area A_W and weighted delay D_W:

$$A_W = \sum_{\text{gate type } u} G(u) \times W_G(u) \tag{2}$$

$$D_W = \sum_{\text{gate type } u} D(u) \times W_D(u). \tag{3}$$

Associated with gate type u, $G(u)$ and $W_G(u)$ return the gate count and weight of each gate. In the critical path of the circuit, $D(u)$ and $W_D(u)$ return the number of traversed gate layers and weight of each layer associated with gate type u.

For the problem caused by different security margins, we use a rule-of-thumb to determine resistance to differential and linear cryptanalysis. For differential cryptanalysis, the number of chosen plaintext pairs to attack a cipher is expected to be in the order of $1/P_d$, where P_d is the maximum differential characteristic probability determined by Theorems 1 and 3. Similarly, to attack a cipher using linear cryptanalysis, the number of known plaintexts is expected to be in the order of $1/P_l$, where P_l is the maximum linear characteristic probability.

Based on above considerations, we define three hardware performance metrics η_s, η_t, and η to measure the space, time, and overall performance, respectively. The three metrics integrate security and complexity and are defined as follows:

$$\eta_s = \frac{\log_2 1/P}{\#\text{ of rounds } \times A_W \text{ per round}} \tag{4}$$

$$\eta_t = \frac{\log_2 1/P}{\#\text{ of rounds } \times D_W \text{ per round}} \tag{5}$$

$$\eta = \frac{\log_2 1/P}{\#\text{ of rounds } \times (A_W \times D_W \text{ per round})} \tag{6}$$

where $P = P_d$ for hardware performance in relation to differential attacks and $P = P_l$ in relation to linear attacks. In each expression, the numerator is essentially a security measure in bits and the denominator is a complexity measure. Since we assume that the S-boxes in the three discussed cipher structures satisfy $p_s = q_s$, the values of $\log_2 1/P_d$ and $\log_2 1/P_l$ are the same. For the nested SPNs and Feistel networks discussed in Section 2, $\log_2 1/P$ is a linear function of the number of rounds. Therefore, the values of η_s, η_t, and η indicate how much security is expected to be obtained for a specific hardware cost, regardless of the number of rounds in a cipher.

Targeted to the same design method, η_s shows the security contribution provided by each area unit; η_t shows the security contribution provided by each delay unit. For a fast implementation such as a pipelined or parallel design, a high η_s means that many independent blocks can be processed simultaneously. For a round iterated design, a high η_t means that the encryption time for a block is small. More generally, using the classical delay-area product as its denominator, η indicates the performance integrating both the delay and area complexities.

The cases that we compare in the following sections are generated as 128-bit block ciphers defined by the nested SPN and Feistel networks. To calculate the gate count and number of gate layers per round, we consider the construction of the combinational circuits of the round structure with S-box and MDS mapping components which can produce high efficiencies in hardware. The hardware design and optimization of these components are described in [6]. The detailed data used in the complexity estimation is presented in the Appendix.

3.1 Hardware Performance of Nested SPNs

A set of nested SPNs can be generated with appropriate configurations of parameterized MDS_L, MDS_H, and S-boxes. As Theorem 2 illustrates, the MDS mapping defined over a large Galois field can be simplified using several mappings in a smaller Galois field. Table 1 lists the cases of nested SPNs in 12 categories (labelled as N1 to N12) defined by the S-boxes and MDS_L. Thus, the cases within a category only differ in the simplification of MDS_H. Each case can be regarded as a 128-bit cipher, after a particular key schedule is defined. Due to the difficulty of finding optimized MDS mappings, the cases with a Galois field larger than GF(2^8) are not considered.

In relation to real ciphers, Case N4-a includes Square, Rijndael, and Anubis. Type II of Hierocrypt belongs to Case N4-b with a simplified MDS_H over GF(2^4). Similar to SHARK [13] and Khazad, Case N8 is a one-level SPN. However, SHARK and Khazad are 64-bit ciphers because their MDS mappings are based on a (16, 8, 9)-code over GF(2^8).

From the viewpoint of implementation, a nested SPN follows the iterative dataflow of key addition, one S-box layer, and an MDS mapping layer (either MDS_L or MDS_H). Since S-boxes cost the most hardware complexity, a 128-bit multiplexor selects MDS_L and MDS_H dynamically such that only one layer of S-boxes is needed in a round iterated design. So assuming a round iterated implementation, the round circuit used for each case in Table 1 includes a 128-bit key addition, one layer of S-boxes, MDS_L, MDS_H, and a

Table 1: 128-bit Nested SPNs of $4r$ Rounds

Case	S-box size	MDS_L : $l_1 \times (2m_1, m_1, m_1{+}1)$ over GF(2^{n_1})	MDS_H : $l_2 \times (2m_2, m_2, m_2{+}1)$ over GF(2^{n_2})	P_d, P_l
N1-a	8×8	$8 \times (4, 2, 3)$ over GF(2^8)	$2 \times (16, 8, 9)$ over GF(2^8)	2^{-162r}
N1-b			$4 \times (16, 8, 9)$ over GF(2^4)	
N2-a	8×8	$16 \times (4, 2, 3)$ over GF(2^4)	$2 \times (16, 8, 9)$ over GF(2^8)	2^{-162r}
N2-b			$4 \times (16, 8, 9)$ over GF(2^4)	
N3-a	8×8	$32 \times (4, 2, 3)$ over GF(2^2)	$2 \times (16, 8, 9)$ over GF(2^8)	2^{-162r}
N3-b			$4 \times (16, 8, 9)$ over GF(2^4)	
N4-a	8×8	$4 \times (8, 4, 5)$ over GF(2^8)	$4 \times (8, 4, 5)$ over GF(2^8)	2^{-150r}
N4-b			$8 \times (8, 4, 5)$ over GF(2^4)	
N5-a	8×8	$8 \times (8, 4, 5)$ over GF(2^4)	$4 \times (8, 4, 5)$ over GF(2^8)	2^{-150r}
N5-b			$8 \times (8, 4, 5)$ over GF(2^4)	
N6-a	8×8	$2 \times (16, 8, 9)$ over GF(2^8)	$8 \times (4, 2, 3)$ over GF(2^8)	2^{-162r}
N6-b			$16 \times (4, 2, 3)$ over GF(2^4)	
N7-a	8×8	$4 \times (16, 8, 9)$ over GF(2^4)	$8 \times (4, 2, 3)$ over GF(2^8)	2^{-162r}
N7-b			$16 \times (4, 2, 3)$ over GF(2^4)	
N7-c			$32 \times (4, 2, 3)$ over GF(2^2)	
N8	8×8	$1 \times (32, 16, 17)$ over GF(2^8)	same as MDS_L	2^{-204r}
N9	4×4	$16 \times (4, 2, 3)$ over GF(2^4)	$1 \times (32, 16, 17)$ over GF(2^8)	2^{-102r}
N10	4×4	$32 \times (4, 2, 3)$ over GF(2^2)	$1 \times (32, 16, 17)$ over GF(2^8)	2^{-102r}
N11-a	4×4	$8 \times (8, 4, 5)$ over GF(2^4)	$2 \times (16, 8, 9)$ over GF(2^8)	2^{-90r}
N11-b			$4 \times (16, 8, 9)$ over GF(2^4)	
N12-a	4×4	$4 \times (16, 8, 9)$ over GF(2^4)	$4 \times (8, 4, 5)$ over GF(2^8)	2^{-90r}
N12-b			$8 \times (8, 4, 5)$ over GF(2^4)	

128-bit multiplexor[2]. The 128-bit multiplexor can be implemented by 385 NAND gates (i.e., $y = x_1 \cdot c + x_2 \cdot \overline{c}$ where c is the select signal and "+" denotes OR).

For the main components and the iterative round structures of each SPN, Table A-1 in the Appendix lists their gate counts and delays of layers. Although each individual value in Table A-1 cannot be perfectly accurate, the comparison of these measures does enable us to distinguish the cases which are more efficient in hardware.

Figure 3 shows the tendency of the universal performance comparison when $W_G(u) = W_D(u) = 1$ for any gate type u (i.e., all gates are assumed to have the same hardware cost). In an ASIC design, XOR gates are more expensive than other gates such as NOT, AND, and OR gates. Figure 4 shows a weighted performance comparison when $W_G(\text{XOR}) = W_D(\text{XOR}) = 2$ and weight for others is one. The two figures follow the similar tendency in performance comparison:

- The size of the S-box largely determines space and time performances. Using small S-boxes tends to cost less hardware area, but more delay than using large S-boxes. Given fixed chip area, the cipher cases using small S-boxes are more advantageous for

[2] MDS Multiplexing is not necessary for N8.

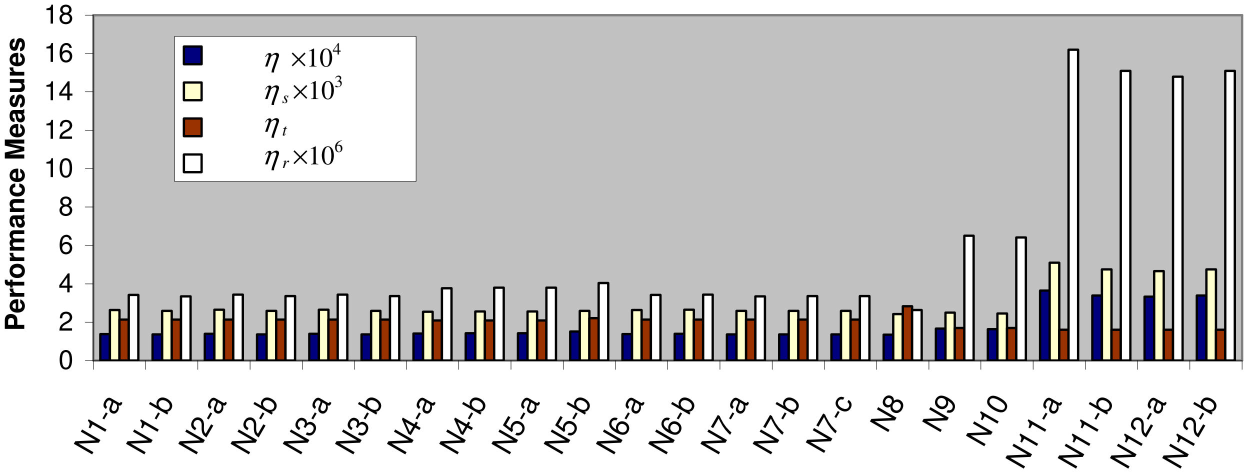


Figure 3: Universal Performance Comparison of Nested SPNs

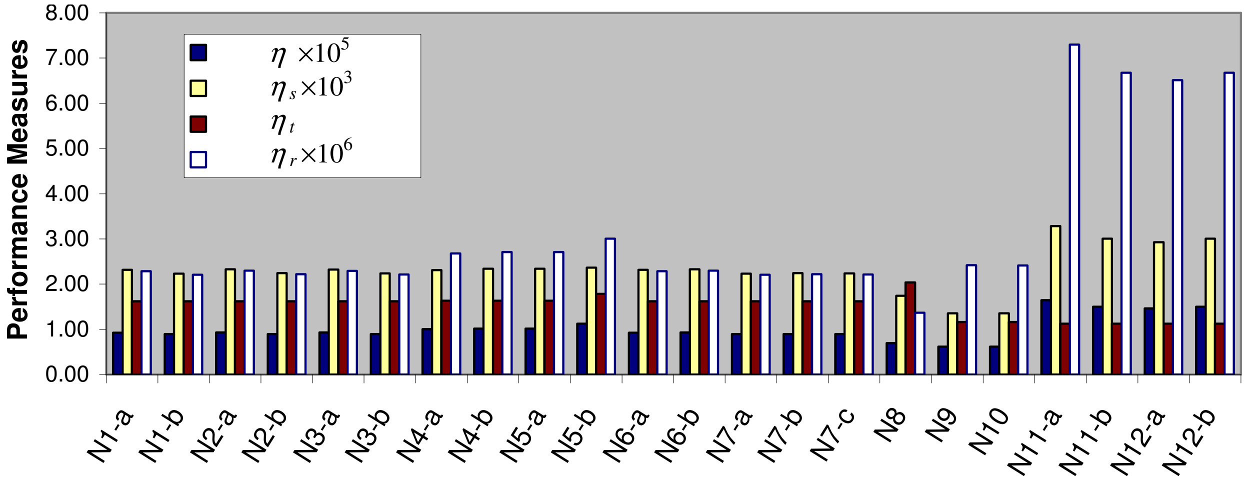


Figure 4: Weighted Performance Comparison of Nested SPNs

parallelism as their higher η_s values show.

- Many SPN structures (N1-N10, N11-N12) are essentially equivalent with respect to their hardware performance. Hence, it is wise for a cipher designer to consider those structures which can facilitate software implementations.

- When the symbol size is 8 bits or less, the simplification of MDS mappings through concatenation does not significantly improve the performance when the MDS mappings have been selected to be optimized for hardware. For example, Case N4-b in Table 1 does not gain a much higher improvement in hardware than Case N4-a.

- When m_1 or m_2 is very high, the MDS mapping determined by m_1 or m_2 (e.g., MDS_H in cases of N9 and N10) will cost much more hardware and overwhelm S-box costs, which degrades the cipher performance.

- As a cipher of Case N4-a, Rijndael is very suitable for a round iterated design. However, its suitability for pipelined or parallel implementations is not as high as cipher cases using 4×4 S-boxes such as cases of N11 and N12.

The above conclusions are based on hardware complexity and security against differential and linear attacks. For some other attacks such as Square attack, the effectiveness significantly decreases after a certain number. In this circumstance, a performance metric of the round structure is defined as:

$$\eta_r = \frac{1}{A_W \times D_W \text{ per round}} .$$

Since the security in bits to resist these attacks increases very rapidly in the number of rounds, with a trend much steeper than differential and linear attacks as more rounds are appended, we take a fixed number of rounds (e.g., about 8 for the Square attack to Rijndael) as enough for the security. The comparison of round performance is also included in Figures 3 and 4. It is obvious that the nested SPNs with small S-boxes and modest sized MDS_L and MDS_H have significantly better performance in relation to the Square attack than other cases.

3.2 Hardware Performance of Feistel Networks

The Feistel network discussed in this section is limited to the subset described in Section 2.4, which has an SPN round function. To construct a typical 128-bit cipher, such a Feistel network has a 64-bit F-function which contains sixteen 4×4 or eight 8×8 parallel S-boxes followed by an MDS mapping layer. As listed in Table 2, six categories (labelled as F1 to F6) of these 128-bit Feistel networks can be generated. To ensure a good avalanche effect, an appropriate fixed permutation of MDS symbols after the MDS mapping is expected, which does not cost any gates. The hardware of one round of the cipher includes a 64-bit XOR for round key addition, one layer of S-boxes, one MDS mapping, and another 64-bit XOR appended to the output of the F-function. The cases of the same category in Table 2 only differ in the simplification of the MDS mapping. The performance comparison in Figures 5 and 6 indicates (refer to the Appendix for detailed data):

- It is useful to pick an MDS mapping that has a large branch number (i.e., $m+1$). The cases with such an MDS mapping have significantly higher values in all three performance measures.

- With high η_t values, the cases with 8×8 S-boxes demonstrate high performance in non-pipelined and non-parallel implementations. With high η_s values, the cases with 4×4 S-boxes demonstrate high performance in pipelined and parallel implementations because many independent blocks can be processed simultaneously.

Camellia is a 128-bit Feistel cipher with a 64-bit round function which consists of eight 8×8 invertible S-boxes and a linear transformation. Hence, Camellia is similar to our discussed Feistel networks but does not use an MDS mapping. The branch number of the Camellia

Table 2: 128-bit Feistel Networks of $4r$ Rounds

Case	S-box size	MDS $l\times(2m,m,m+1)$ over GF(2^n)	P_d, P_l
F1-a	8×8	$4\times(4,2,3)$ over GF(2^8)	$2^{-6(3r+\lfloor\frac{r}{2}\rfloor)}$
F1-b		$8\times(4,2,3)$ over GF(2^4)	
F1-c		$16\times(4,2,3)$ over GF(2^2)	
F2-a	8×8	$2\times(8,4,5)$ over GF(2^8)	$2^{-6(5r+\lfloor\frac{r}{2}\rfloor)}$
F2-b		$4\times(8,4,5)$ over GF(2^4)	
F3-a	8×8	$1\times(16,8,9)$ over GF(2^8)	$2^{-6(9r+\lfloor\frac{r}{2}\rfloor)}$
F3-b		$2\times(16,8,9)$ over GF(2^4)	
F4-a	4×4	$4\times(4,2,3)$ over GF(2^8)	$2^{-2(3r+\lfloor\frac{r}{2}\rfloor)}$
F4-b		$8\times(4,2,3)$ over GF(2^4)	
F4-c		$16\times(4,2,3)$ over GF(2^2)	
F5-a	4×4	$2\times(8,4,5)$ over GF(2^8)	$2^{-2(5r+\lfloor\frac{r}{2}\rfloor)}$
F5-b		$4\times(8,4,5)$ over GF(2^4)	
F6-a	4×4	$1\times(16,8,9)$ over GF(2^8)	$2^{-2(9r+\lfloor\frac{r}{2}\rfloor)}$
F6-b		$2\times(16,8,9)$ over GF(2^4)	

linear transformation is 5. An efficient implementation of such a linear transformation costs 176 two-input XOR gates and a delay of 3 gate layers in universal comparison. Thus, Camellia has performance similar to Case F2-a which has 264 XOR gates and a delay of 3 gate layers (see Table A-2 in the Appendix). Compared with the case F3-a, Camellia has a slightly more compact round structure (i.e., about 5% less in gate count than Case F3-a). However, each round of Camellia contributes much less to the security. Eleven rounds of F3-a provides equivalent security to nineteen rounds of Camellia. Further calculation shows that the overall hardware performance of F3-a is about 50% higher than that of Camellia. The weighted performance comparison follows a similar trend.

3.3 Synthesis Results

The above performance analysis is based on theoretical evaluation of hardware complexity. The usability of these analytical results can be verified when VLSI technology is targeted. To avoid arduous work on synthesizing each cipher case, we did a high level synthesis of each component used in Tables 1 and 2. The components are coded in VHDL and synthesized with Synopsys Design Compiler. Two CMOS libraries[3] were used where most standard cells have one or two bitwise inputs.

During synthesis, if the minimum area (respectively, delay) is set as the main constraint[4], the numbers of equivalent gates (respectively, critical delay time) of 8×8 S-boxes are close to their estimates in Tables A-1 and A-2. The gates and delays of 4×4 S-boxes are slightly less than their estimates because it is much easier for CAD tools to simplify smaller S-boxes. This effect indicates that the performance advantage of using small S-boxes as shown in

[3] lsi_10k.db and TSMC's 0.18μm CMOS library are targeted separately.

[4] When other constraint is set, the absolute values of area and delay will vary, but their comparison trend follows a similar trend.

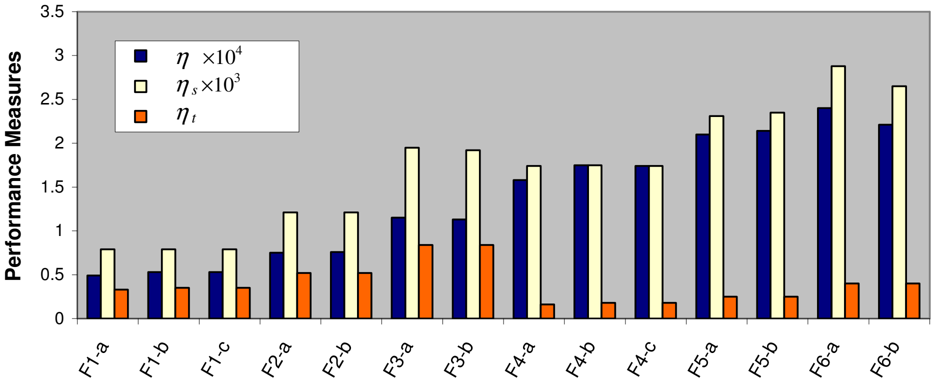


Figure 5: Universal Performance Comparison of Feistel Networks

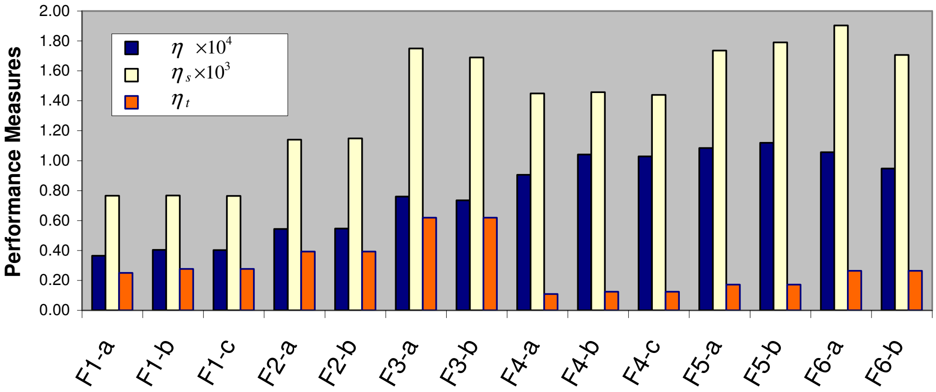


Figure 6: Weighted Performance Comparison of Feistel Networks

Figures 3 to 6 is decent and slightly understated.

Since the MDS mapping is implemented in XOR gates, the areas and delays closely follow the proportional relation of their estimations in Tables A-1 and A-2. Because XOR gates are larger and slower than other gate types, synthesis tools may replace them with other gates such as NXORs during optimization. Nevertheless, the delays and numbers of equivalent gates imply that a weight of 2 is reasonable for an XOR gate. This effect makes the cases with large MDS mapping worse in weighted performance, e.g., the cases in N8 to N12, F5, and F6. This problem is encountered in the realizations where a large percent of XORs are used. The weighted performance shown in Figures 4 and 6 are thus more useful for a closer comparison than the universal method.

4 Conclusions

In this paper we have considered two cipher structures composed of S-boxes and MDS mappings. Various cipher cases are generated from these structures with different component configurations. Their security and complexity are examined and integrated into performance metrics.

In hardware, the discussed cipher cases using large S-boxes are suitable for non-pipelined and non-parallel applications where delay is the main design criterion; however, in pipelined and parallel applications, the cipher cases using small S-boxes produce high performance. Further, appropriate selection of an MDS mapping layer is important for security against differential and linear attacks.

Compared with Feistel networks, the nested SPNs generally have higher hardware performance. When the same S-boxes are used, a nested SPN tends to be more efficient in hardware to resist differential and linear attacks. Considering the threat of Square attacks, nested SPNs with smaller S-boxes are preferred. For a Feistel network, more rounds are needed to be secure against differential and linear attacks. With little change in the linear transformation, a suggestion is made to improve Camellia in terms of security and hardware efficiency.

In line with a nested SPN, MISTY [19] can be regarded as a nested Feistel network. Using provable security as the security measure, it will be interesting future work to compare the hardware performance between these two nested structures with similar performance metrics defined in Section 3.

References

[1] National Institute of Standards and Technology, "Data Encryption Standard (DES)", *Federal Information Processing Standard 46*, 1977.

[2] J. Daemen and V. Rijmen, "AES Proposal: Rijndael", *Advanced Encryption Standard*, available on: csrc.nist.gov/encryption/aes/rijndael.

[3] K. Ohkuma, H. Muratani, F. Sano, and S. Kawamura, "The Block Cipher Hierocrypt", *Workshop on Selected Areas in Cryptography - SAC 2000*, Lecture Notes in Computer Science 2012, Springer-Verlag, pp. 72-88, 2001.

[4] P. Barreto and V. Rijmen, "The Anubis Block Cipher", *NESSIE Algorithm Submission*, 2000, available on: www.cosic.esat.kuleuven.ac.be/nessie.

[5] P. Barreto and V. Rijmen, "The Khazad Legacy-Level Block Cipher", *NESSIE Algorithm Submission*, 2000, available on: www.cosic.esat.kuleuven.ac.be/nessie.

[6] L. Xiao and H.M. Heys, "Hardware Design and Analysis of Block Cipher Components", accepted for presentation at the *5th International Conference on Information Security and Cryptology - ICISC 2002*, Seoul, Korea, November 28-29, 2002.

[7] K. Aoki, T. Ichikawa, M. Kanda, M. Matsui, S. Moriai, J. Nakajima, and T. Tokita, "Camellia: A 128-bit Block Cipher Suitable for Multiple Platforms", *NESSIE Algorithm Submission*, 2000, available on: www.cosic.esat.kuleuven.ac.be/nessie.

[8] E. Biham and A. Shamir, “Differential cryptanalysis of DES-like cryptosystems”, *Advances in Cryptology - CRYPTO ’90* , Lecture Notes in Computer Science 537, pp. 2-21. Springer-Verlag, 1991.

[9] M. Matsui, “Linear Cryptanalysis Method for DES Cipher”, *Advances in Cryptology - Eurocrypt ’93*, Lecture Notes in Computer Science 765, Springer-Verlag, pp. 386-397, 1993.

[10] R. Anderson, E. Biham and L. Knudsen, “Serpent: A Proposal for the Advanced Encryption Standard”, *AES Algorithm Submission*, available on: www.cl.cam.ac.uk/∼rja14/serpent.html

[11] Toshiba Corporation, “Security Evaluation: Hierocrypt-3”, *NESSIE Algorithm Submission*, 2000, available on: www.cosic.esat.kuleuven.ac.be/nessie.

[12] F.J. MacWilliams and N.J.A. Sloane, *The Theory of Error-Correcting Codes*, North-Holland, Amsterdam, 1977.

[13] V. Rijmen, J. Daemen, B. Preneel, A. Bosselaers, and E. De Win, “The cipher SHARK”, *Workshop on Fast Software Encryption - FSE ’96*, Lecture Notes in Computer Science 1039, Springer-Verlag, pp. 99-112, 1997.

[14] J. Daemen, L. R. Knudsen, and V. Rijmen, “The Block Cipher Square”, *Workshop on Fast Software Encryption - FSE ’97*, Lecture Notes in Computer Science 1267, Springer-Verlag, pp. 54-68, 1997.

[15] M. Kanda, Y. Takashima, T. Matsumoto, K. Aoki, and K.Ohta, “ Strategy for Constructing Fast Round Functions with Practical Security Against Differential and Linear Cryptanalysis”, *Workshop on Selected Areas in Cryptography - SAC ’98*, Lecture Notes in Computer Science 1556 , pp. 264-279, 1999.

[16] M. Kanda, “Practical Security Evaluation against Differential and Linear Attacks for Feistel Ciphers with SPN Round Function”, *Workshop on Selected Areas in Cryptography - SAC 2000*, Lecture Notes in Computer Science 2012, Springer-Verlag, pp. 324-338, 2001.

[17] C. Paar, “Efficient VLSI Architectures for Bit-Parallel Computation in Galois Fields”, Ph.D. Thesis, Institute for Experimental Mathematics, University of Essen, Germany, 1994.

[18] J. Nechvatal, E. Barker, L. Bassham, W. Burr, M. Dworkin, J. Foti, and E. Roback, “Report on the Development of the Advanced Encryption Standard (AES)”, *Report on the AES Selection from U.S. National Institute of Standards and Technology (NIST)*, available on: csrc.nist.gov/encryption/aes.

[19] M. Matsui, “New Block Encryption Algorithm MISTY”, *Workshop on Fast Software Encryption - FSE ’97*, Lecture Notes in Computer Science 1267, Springer-Verlag, pp. 54-68, 1997.

Appendix: Complexity Evaluation of Cipher Components

In hardware, the complexity of S-boxes are evaluated through the simplification results deduced from an encoder-switch-decoder model [6]. In this model, S-boxes are composed of low complexity gates (ANDs, ORs, and NOTs). A 4×4 S-box can be implemented using 50 gates and produces a delay of 6 gate layers; an 8×8 S-box can be implemented using 806 gates and produces a delay of 11 gate layers. Involution MDS codes [4] are found by searching Hadamard matrices and have been optimized for hardware [6]. MDS codes are composed of XORs. The evaluated hardware costs of S-boxes, MDS mappings, and round structures are listed in Tables A-1 and A-2 for each cipher case. The values of performance metrics in the two tables are calculated for universal comparison only.

Using these results, the complexity of each 128-bit 2-level nested SPN is evaluated for each round. The hardware of one round SPN includes a 128-bit key addition layer, an S-box layer, two MDS mappings at different levels, and a 128-bit multiplexor. The 128-bit multiplexor selects MDS_L and MDS_H alternatively in consecutive rounds, which costs 385 NAND gates and a delay of two gate layers. The key addition costs 128 XOR gates and a delay of one gate level. The calculation of the delay per round assumes the highest delay of MDS_L and MDS_H.

The hardware of one round of the Feistel network includes a 64-bit key addition layer, an S-box layer, an MDS mapping layer, and a 64-bit XOR after the F-function (as shown in Figure 2(a)). The key addition costs 64 XOR gates and a delay of one gate level. The XOR after the F-function has the same hardware complexity as the key addition.

Table A-1: Complexity and Universal Performance Estimation of One Round of 128-bit Nested Involution SPNs in Hardware

Case	S-boxes	MDS_L	MDS_H	Round Total (universal)	η_s	η_t	η	η_r
	Gate#/Delay	XOR#/Delay	XOR#/Delay	Gate#/Delay	(10^{-3})		(10^{-4})	(10^{-6})
N1-a	12896 / 11	256 / 3	1728 / 5	15393 / 19	2.63	2.13	1.38	3.42
N1-b	12896 / 11	256 / 3	2048 / 5	15713 / 19	2.58	2.13	1.36	3.35
N2-a	12896 / 11	208 / 2	1728 / 5	15345 / 19	2.64	2.13	1.39	3.43
N2-b	12896 / 11	208 / 2	2048 / 5	15665 / 19	2.59	2.13	1.36	3.36
N3-a	12896 / 11	224 / 2	1728 / 5	15361 / 19	2.64	2.13	1.39	3.43
N3-b	12896 / 11	224 / 2	2048 / 5	15681 / 19	2.58	2.13	1.36	3.36
N4-a	12896 / 11	672 / 4	672 / 4	14753 / 18	2.54	2.08	1.41	3.77
N4-b	12896 / 11	672 / 4	576 / 3	14657 / 18	2.56	2.08	1.42	3.79
N5-a	12896 / 11	576 / 3	672 / 4	14657 / 18	2.56	2.08	1.42	3.79
N5-b	12896 / 11	576 / 3	576 / 3	14561 / 17	2.58	2.21	1.51	4.04
N6-a	12896 / 11	1728 / 5	256 / 3	15393 / 19	2.63	2.13	1.38	3.42
N6-b	12896 / 11	1728 / 5	208 / 2	15345 / 19	2.64	2.13	1.39	3.43
N7-a	12896 / 11	2048 / 5	256 / 3	15713 / 19	2.58	2.13	1.36	3.35
N7-b	12896 / 11	2048 / 5	208 / 2	15665 / 19	2.59	2.13	1.36	3.36
N7-c	12896 / 11	2048 / 5	224 / 2	15681 / 19	2.58	2.13	1.36	3.36
N8	12896 / 11	8064 / 6	8064 / 6	21088 / 18	2.42	2.83	1.34	2.63
N9	1600 / 6	208 / 2	8064 / 6	10257 / 15	2.49	1.70	1.66	6.50
N10	1600 / 6	224 / 2	8064 / 6	10401 / 15	2.45	1.70	1.63	6.41
N11-a	1600 / 6	576 / 3	1728 / 5	4417 / 14	5.09	1.61	3.64	16.2
N11-b	1600 / 6	576 / 3	2048 / 5	4737 / 14	4.75	1.61	3.39	15.1
N12-a	1600 / 6	2048 / 5	672 / 4	4833 / 14	4.66	1.61	3.33	14.8
N12-b	1600 / 6	2048 / 5	576 / 3	4737 / 14	4.75	1.61	3.39	15.1

Table A-2: Complexity and Universal Performance Estimation of One Round of 128-bit Feistel Networks in Hardware

Case	S-boxes	MDS	Round Total (universal)	η_s	η_t	η
	Gate # / Delay	XOR # / Delay	Gate # / Delay	(10^{-3})		(10^{-4})
F1-a	6448 / 11	76 / 3	6652 / 16	0.79	0.33	0.49
F1-b	6448 / 11	72 / 2	6648 / 15	0.79	0.35	0.53
F1-c	6448 / 11	80 / 2	6656 / 15	0.79	0.35	0.53
F2-a	6448 / 11	264 / 3	6840 / 16	1.21	0.52	0.75
F2-b	6448 / 11	240 / 3	6816 / 16	1.21	0.52	0.76
F3-a	6448 / 11	720 / 4	7296 / 17	1.95	0.84	1.15
F3-b	6448 / 11	864 / 4	7440 / 17	1.92	0.84	1.13
F4-a	800 / 6	76 / 3	1004 / 11	1.74	0.16	1.58
F4-b	800 / 6	72 / 2	1000 / 10	1.75	0.18	1.75
F4-c	800 / 6	80 / 2	1008 / 10	1.74	0.18	1.74
F5-a	800 / 6	264 / 3	1192 / 11	2.31	0.25	2.10
F5-b	800 / 6	240 / 3	1168 / 11	2.35	0.25	2.14
F6-a	800 / 6	720 / 4	1648 / 12	2.88	0.40	2.40
F6-b	800 / 6	864 / 4	1792 / 12	2.65	0.40	2.21