DESIGN AND IMPLEMENTATION OF AN ADVANCED SUBSTITUTION-PERMUTATION ENCRYPTION NETWORK

JIANHONG XU

Your file Votre référence
ISBN: 978-0-494-42087-4
Our file Notre référence
ISBN: 978-0-494-42087-4

Design and Implementation of an Advanced Substitution-Permutation Encryption Network

by

©Jianhong Xu

A thesis submitted to the
School of Graduate Studies
in partial fulfillment of the requirements for
the degree of Master of Engineering

Faculty of Engineering and Applied Science
Memorial University of Newfoundland

August, 1997

St. John's Newfoundland Canada

Abstract

To solve the problems of data security in modern electronic communication environments and applications, researchers have been placing much effort on the design of efficient and secure ciphers. Substitution-permutation encryption networks (SPNs) are an important class of private-key block ciphers. The objective of this thesis is to develop an advanced substitution-permutation encryption network that not only is efficiently secure but also can be simply implemented in both hardware and software.

Two of the most powerful attacks are linear cryptanalysis and differential cryptanalysis. After investigating the application of linear cryptanalysis to an SPN, a new nonlinearity criterion for the design of S-boxes is presented. S-boxes satisfying this criterion strengthen the ability of an SPN to frustrate linear cryptanalysis. As well, we propose a novel linear transformation as the method of interconnection between rounds of S-boxes. The use of the linear transformation increases the resistance of an SPN to both linear cryptanalysis and differential cryptanalysis.

Finally, we implement an SPN which consists of our new linear transformation and 4×4 S-boxes satisfying our new design criterion by using a Field Programmable Gate Array (FPGA). The simulation results confirm that the digital hardware implementation of the SPN is practical and not complicated.

Dedication

To my parents

Acknowledgments

I would like to especially thank my supervisor, Dr. Howard Heys, whose wisdom, enthusiasm, and patience in instructing me throughout the course of this thesis will never be forgotten.

I also gratefully acknowledge the financial support of the Faculty of Engineering and Applied Science of Memorial University of Newfoundland.

Contents

List of Figures

List of Tables

NOTATION

Variables and Mathematical Conventions

N	Cipher block size
n	Bijective S-box input/output size
R	Number of substitution rounds in cipher
r	Substitution round number
S	A substitution box (S-box)
$P = [p_1\ p_2 \ldots p_N]$	Plaintext vector
$C = [C_1\ C_2 \ldots C_N]$	Ciphertext vector
$K = [K_1\ K_2 \ldots K_r]$	Cipher key vector
λ	Diffusion order
$wt(U)$	Hamming weight of vector U
$U_i \oplus V_j$	Exclusive-OR (XOR) of bits U_i and V_j
$\Delta U = U' \oplus U''$	XOR difference of two values, U' and U'', of vector U

Chapter 1

Introduction

Every day, millions of people use telephones, fax machines, and computer networks to exchange information. Electronic communication is now an unavoidable component of modern life. Ensuring communications security appears to be more and more important. As cryptography is seen as the only effective means of ensuring security in communications and in computer systems, increased research effort is now being applied to the area of cryptography.

Cryptography is the science of techniques which make information unintelligible and unmodifiable by outsiders and still comprehensible or verifiable by the intended receiver. A cryptographic system or cryptosystem is referred to as any system which applies methods of cryptography to transform data and restore data. A general cryptosystem is shown in Figure 1.1. Encryption is a special computation that operates on messages, converting them into a representation that is in unintelligible form. The original message is called plaintext and the transformed representation is called ciphertext. Decryption is used to reverse the process of encryption: it accepts ciphertext as input and yields the corresponding plaintext. Both encryption and decryption are controlled by a key, which is a parameter to the process. It should be beyond the means of the eavesdropper, who has no access to the key of the receiver, to obtain the plaintext from the ciphertext.

There are two general forms of cryptographic algorithms or ciphers: private-key and public-key. In a private-key cipher, the same key is used to encrypt and decrypt data. (In Figure 1.1, the sender's key (key 1) and the receiver's key (key 2) are secret and identical). In a public-key cipher the mathematically related but different keys are used for encryption and decryption.

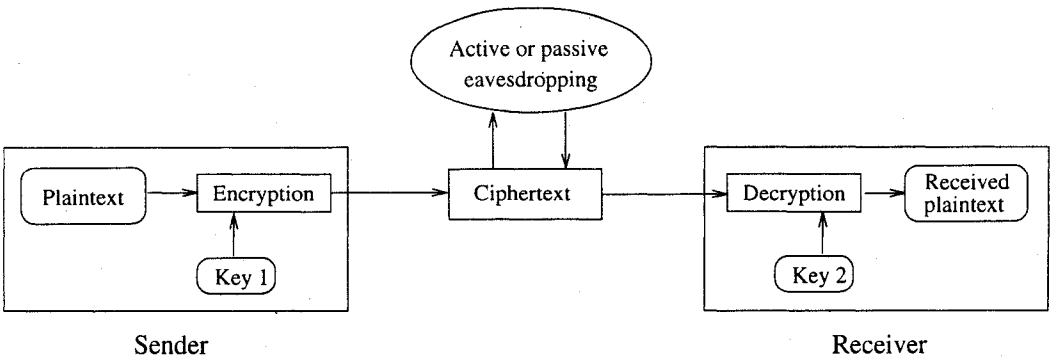


Figure 1.1: A General Cryptographic System

One key is kept secret and is only known to its owner, whereas the other key is made publicly known. In Figure 1.1, key 1 is different from key 2. The intended receiver can make his key 1 public for all those who want to send him messages and keep his key 2 secret in order to decrypt his messages.

Private-key ciphers can be divided into two categories. Ciphers that operate on the plaintext a single bit at a time are called stream ciphers; ciphers that operate on the plaintext in blocks of bits are called block ciphers.

In this thesis, we examine substitution-permutation networks or SPNs, which are a class of private-key block ciphers. Shannon [24] proposed using the concepts of "confusion" and "diffusion" to create a mixing transformation, which uniformly distributes the redundant statistical properties of the plaintext over the set of all possible ciphertexts. "Confusion" means that the relationship between input and output is mathematically complex. "Diffusion" involves spreading local effects in input across all output bits. The SPN architecture first proposed by Feistel [9] is directly based on the principles of confusion and diffusion. A basic SPN uses small substitutions, called S-boxes, to achieve confusion, and permutations to achieve diffusion.

As the results of the thesis, we first proposed a new nonlinearity criterion for the design of S-boxes. S-boxes satisfying this criterion and the diffusion order requirement improve remarkably the ability of an SPN to resist linear cryptanalysis and differential cryptanalysis. Secondly, we designed a new linear transformation as the method of interconnection between rounds of S-boxes. When the linear transformation is adopted in an SPN, the ability of an SPN to resist linear cryptanalysis and differential cryptanalysis is strengthened noticeably. Finally, we

implemented an SPN constructed from our new 4×4 S-boxes and our novel linear transformation using a Field Programmable Gate Array (FPGA). The information about the complexity of the FPGA implementation shows that the digital hardware implementeation of our SPN is practical and not complicated.

1.1 Motivation for the Research

Designing efficient, secure ciphers that keep pace with modern electronic communication environments and applications motivated us to do this research. The Data Encryption Standard (DES) [17], which has been by far the most popular private-key block cipher used, will soon be unusable for securing modern electronic communications due to its inadequacies. The size of the keyspace, 2^{56}, is too small to be secure with the speed of today's computer hardware. According to Wiener's design [28], a $1,000,000 machine could search the entire key space in about 3.5 hours. Also, the secrecy surrounding the design of the DES algorithm has caused suspicion that the National Security Agency embedded a "trapdoor" into the cryptosysytem and prevented any simple modifications or extensions of the algorithm. To date, no ciphers have been well enough developed such that they would convincingly be able to face the challenge of modern electronic technology and replace DES. The National Institute of Standards and Technology (NIST) is currently calling for cipher proposals [7].

An SPN is a simple yet efficient implementation of a block cipher. Each round of a basic SPN consists of a layer of $n \times n$ S-boxes and a permutation which connects two adjacent layers of S-boxes. The simplicity of its structure is advantageous as it allows us to analyze its strength against various kinds of cryptanalysis and then to effectively improve the cipher. As well, since in an SPN, substitutions and permutations are operated on the whole block of the messages (unlike DES which operates on half a block at a time), intuitively an SPN is more efficiently secure in enciphering a message. That is, an SPN with a small number of rounds of operation should attain a great security. These two basic properties of an SPN give us confidence to select an SPN as the object of study that can reasonably be designed to meet the security needs of modern electronic communications.

1.2 Outline of the Thesis

This thesis is organized as follows:

- Chapter 2 provides an introduction to the substitution-permutation encryption network. Some techniques of cryptanalysis related to this thesis are also introduced.
- Chapter 3 presents a new criterion for the design of 8×8 S-boxes.
- Chapter 4 introduces a novel linear transformation and shows its ability in helping an SPN against linear cryptanalysis.
- Chapter 5 examines the capacity of the linear transformation in thwarting differential cryptanalysis.
- Chapter 6 discusses the implementation of our SPN using a Field Programmable Gate Array (FPGA).
- Chapter 7 provides a summary of the thesis and proposals for future work.

Chapter 2

Background

In this chapter, the basic knowledge necessary for understanding the substitution-permutation network is presented. Some cryptanalysis techniques related to this thesis are also introduced. As well, previous research related to the design of the substitution-permutation network is reviewed.

2.1 Substitution-Permutation Network

The detailed architectures for block ciphers based on Shannon's [24] concept of "confusion" and "diffusion" were first designed by Feistel [9] and Feistel, Notz, and Smith [10]. One of the block cipher architectures became the network structure for substitution-permutation networks (SPN), and the other became that for DES-like ciphers.

Both an SPN and a DES-like cipher are product ciphers, which iteratively perform simple basic cryptographic operations on the data for a number of rounds. The main difference between the SPN architecture and the DES-like architecture is that an SPN performs substitutions and permutations on the whole block of data, while a DES-like cipher performs these operations on only half the block at a time.

One round of a DES-like cipher is illustrated in Figure 2.1. The round operation of round r can be described as follows:

$$\begin{aligned} \mathbf{R}_{r+1} &= f(\mathbf{R}_r, \mathbf{K}_r) \oplus \mathbf{L}_r \\ \mathbf{L}_{r+1} &= \mathbf{R}_r \end{aligned} \tag{2.1}$$

where $\mathbf{L}_r$ is the left half-block of data, $\mathbf{R}_r$ is the right half block of data, $\mathbf{K}_r$ is the key bits

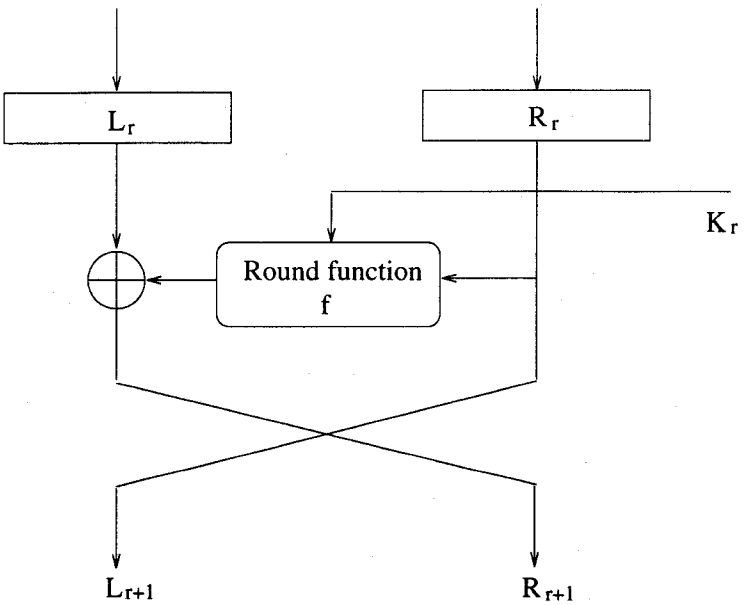


Figure 2.1: One Round of A DES-like Cipher

associated with round r, and f is a function that executes the substitutions and transposition (or permutation).

One round of a basic SPN is shown in Figure 2.2. Each round consists of a layer of substitutions on small sub-blocks and a bit position transposition (permutation). Each of the substitutions is referred as an S-box. As the study in the design of the SPN has been furthered, it has been shown that the permutation can be viewed more generally as a linear transformation [12]. In this thesis, we shall use the S-box interconnection layer to refer to both the permutation and the linear transformation. We shall still refer to these networks as SPNs even though the S-box interconnection layer could be a linear transformation which is not a permutation. Keying the SPN is omitted in Figure 2.2.

Generally, the SPN can be viewed as consisting of three components: S-boxes, S-box interconnections, and key scheduling. We shall elaborate on the notion of these three components.

2.1.1 S-box

In general, an $m \times n$ S-box substitutes an n-bit output block for an m-bit input block. The S-boxes of the SPN must be symmetric (size $n \times n$) and invertible so that an SPN performs a one-to-one mapping and the ciphertext is decryptable. An $n \times n$ symmetric S-box can be viewed

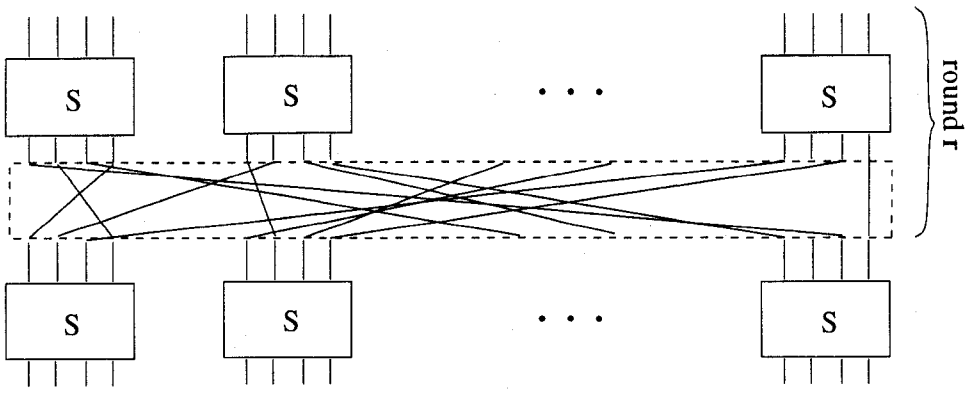
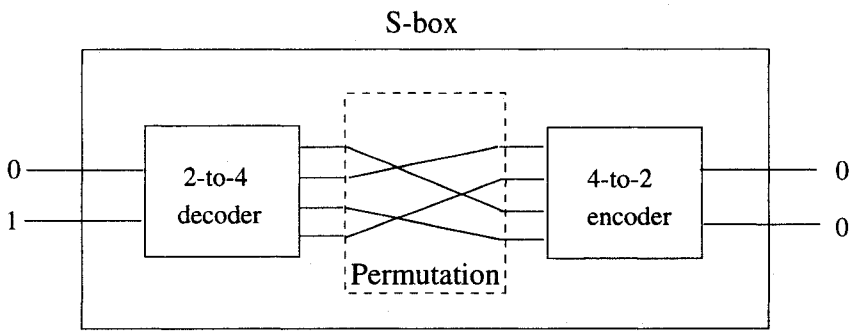


Figure 2.2: One Round of A Basic SPN



IN	OUT
00	10
01	00
10	11
11	01

Figure 2.3: An S-box Example

as formed by sandwiching a permutation between a decoder and a corresponding encoder. It supplies 2^n internal terminals that can be connected in $2^n!$ different ways. Hence, there are a total of $2^n!$ S-boxes of size $n \times n$. A 2×2 symmetric S-box is depicted in Figure 2.3.

In an SPN only the S-boxes can be designed to be nonlinear. A nonlinear S-box implies that for the S-box there does not exist a mod-2 linear combination of input bits such that for all possible input vectors it can be equal to a particular mod-2 linear combination of output bits. The cryptographic strength of an SPN relies highly on the strength of the S-boxes. As a result, the design and analysis of S-boxes has been a topic of considerable interest in the cryptographic community.

Kam and Davida [13] proposed the completeness criterion, which requires that each output bit

of an S-box or cipher depends on every input bit. Webster and Tavares [27] suggested the strict avalanche criterion (SAC) for the design of S-boxes. An S-box satisfies SAC if, over all possible input vectors, inverting input bit i causes output bit j to be changed with a probability of $1/2$ for all i and j. Adams and Tavares [1] proposed a design procedure for the S-boxes which satisfy (i) bijection, (ii) nonlinearity, (iii) strict avalanche criterion, and (iv) output bit independence. (Output bit independence ensures that over all possible input vectors any pair of two output bits are not equal to each other significantly more, or significantly less, than half the time.) According to O'Connor's analysis [5], this procedure becomes impractical as the size of an S-box increases.

Dawson and Tavares [6] extended the work of Forré [11] in applying information theory to S-box design. They suggest selecting S-boxes such that (i) the mutual information between a subset of output bits and any subset of input and/or output bits is minimal, and (ii) the mutual information between a subset of output bit changes and any subset of changes of input and/or output bits is minimal.

Nyberg [20] proposed using bent functions to construct highly nonlinear S-boxes. Unfortunately, the bent functions must be modified to achieve other cryptographically desirable properties such as balance, and there is no known effective way to do so. (Balance implies that the bit which is a Boolean function of input bits takes on the values 0 and 1 equally.)

Heys and Tavares [12] stated that (i) large S-boxes are more likely to have high nonlinearity than small S-boxes, and (ii) S-boxes with good diffusion of bit changes increase resistance to differential cryptanalysis. An S-box posseses good diffusion if a one bit input change causes several output bits to change.

While so much research has been conducted into the design of S-boxes, the design of S-boxes that can be easily implemented in hardware and simultaneously have provably good cryptography properties has been largely neglected.

2.1.2 S-box Interconnection

A basic SPN uses permutations as S-box interconnections. A permutation essentially involves reordering the bits in the data. Compared with the work on the design of S-boxes, less work

has done on the design of S-box interconnections.

In [13], Kam and Davida presented a permutation design approach to achieve the completeness property of an SPN. In [2], Ayoub extended the work of Kam and Davida and suggested a class of cryptographically equivalent permutations (CEP) such that (i) the same permutation is used in every round of an SPN, and (ii) the completeness of an SPN can be achieved in the minimum number of rounds. To date Ayoub's permutations seem to be the most practical permutations to employ in an SPN.

Generally, in a linear transformation every output bit is a mod-2 linear combination of some or all of the input bits. A permutation can be viewed as a special case of a linear transformation. Linear transformations used as interconnections in an SPN must be invertible to make decryption feasible.

In [12], Heys and Tavares first suggested using linear transformations between rounds of S-boxes to intensify the quick diffusion of bits and increase the resistance to differential and linear cryptanalysis. In [32], the authors extended the work in [12] and suggested an improved linear transformation which is more effective in resisting differential and linear cryptanalysis.

2.1.3 Keying the Network

In an SPN, two methods (see Figure 2.4) can be used in keying an S-box:

- *Selection Keying*: one or more of the sub-key bits are used to select which of the S-box mappings is to be used.
- *XOR Keying*: the data bits are XORed with the sub-key bits before entering an S-box.

For an SPN using XOR keying, the key must be XORed after last round as well as at the inputs to each round. The reason that the key must be XORed after the last round is that, only after the output of the last round is XORed with the key, the input of the S-boxes in the last round is unknown, so that the last round is a valid round.

During encryption a cipher key is picked for the operation such that it is only known to the sender and receiver. The sub-keys for rounds of an SPN are derived from this cipher key. A

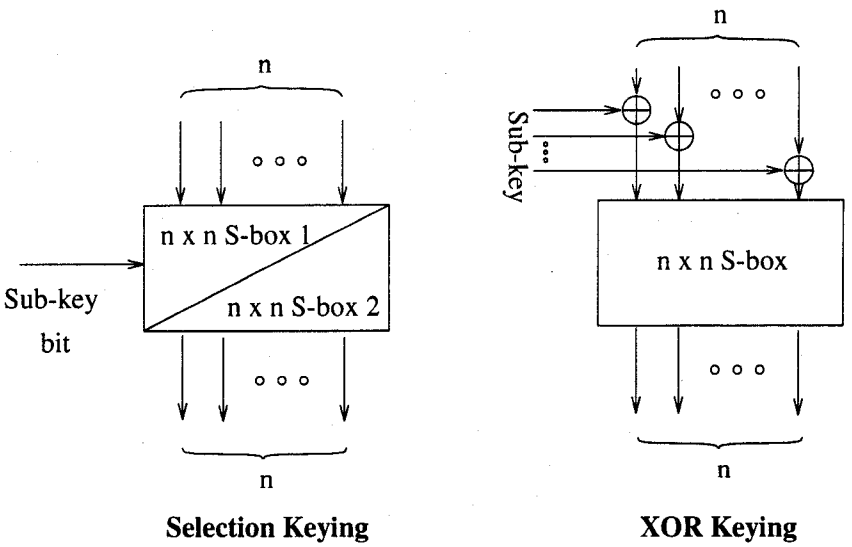


Figure 2.4: Two Keying Methods

different round may have a different sub-key. Determination of the sub-keys from the cipher key is done by a key scheduling algorithm. For an SPN, decryption is accomplished by passing the data in reverse direction through the network. Correspondingly, for decryption, the key schedule must be applied in reverse to encryption.

A key scheduling algorithm easily implemented for an SPN is to apply the cipher key to each round as the sub-key of the round. The security of such a simple key schedule scheme has never been determined.

In this thesis, we will not consider the design of the key scheduling algorithm.

2.2 Other Block Ciphers

To date many block ciphers have been proposed. Some of the most notable ones are FEAL, IDEA, and RC5.

FEAL [25] developed by Shimizu and Miyaguchi is a block cipher similar to DES. It uses a 64-bit data block and a 64-bit key. The basic design principle is to use stronger round functions to reduce the number of rounds, making the algorithm run faster.

IDEA [15] is proposed by Lai and Massey. It is a block cipher which operates on 64-bit plaintexts. The length of the key is 128 bit. The design idea of the cipher is to use the

operations from three different algebraic groups to achieve the efficient mixing transformation. The three algebraic groups are: XOR, addition modulo 2^{16}, and multiplication modulo $2^{16}+1$.

RC5 [22, 23] is a block algorithm designed by Ron Rivest and analyzed by RSA Laboratories. RC5 is in fact a family of algorithms and has three parameters: block size, key size, and number of rounds. Three operations are adopted in the cipher: XOR, addition, and rotation. Since these operations are generally found on most processors, RC5 is expected to be a very fast, secure cipher, easily implemented in software.

2.3 Cryptanalysis Techniques

A cryptosystem is said to be compromised via cryptanalysis if it is possible to recover the plaintext of a message from the ciphertext without knowledge of the key used in the encryption algorithm or if it is possible to derive the key from a set of available ciphertexts or plaintext/ciphertext pairs. It is usually assumed that in cryptanalysis the cryptanalyst knows the details of the cryptosystem. Since a one-time pad cipher is the only scheme that can be proven to be unconditionally secure [26], in modern cryptology a cipher is considered to be unbreakable if the cipher is computationally secure. A cryptosystem is computationally secure if the best known algorithm for breaking it requires an unreasonably large amount of computing time with available resources.

An attack refers to an intended cryptanalysis. There are three general types of cryptanalytic attacks of block ciphers: (1) ciphertext only, (2) known plaintext, and (3) chosen plaintext. In a ciphertext only attack, the cryptanalyst only posseses some ciphertexts. A known plaintext attack assumes that the cryptanalyst obtains the plaintexts corresponding to some known ciphertexts. A chosen plaintext attack is more powerful than a known plaintext attack. In this attack, the cryptanalyst can use some chosen plaintexts to obtain the desired ciphertexts.

The exhaustive key search attack tries every possible key to break a cipher. Ciphers are considered theoretically secure against a particular attack if the attack has a computational complexity that is at least as large as an exhaustive key search. Besides the exhaustive key search attack, the two notable and most powerful attacks against block ciphers are differential cryptanalysis and linear cryptanalysis.

2.3.1 Differential Cryptanalysis

Biham and Shamir [3] discovered the technique of differential cryptanalysis to attack block ciphers. With this method they have attacked a number of block ciphers such as DES [3] and FEAL [4].

Differential cryptanalysis is a chosen plaintext attack which examines the bitwise XORs of pairs of plaintexts and the XORs of the corresponding pairs of ciphertexts. The XOR of a pair of plaintexts is particularly chosen such that a specified sequence of XORs through the rounds of encryption will occur with a relatively high probability. An r-round characteristic is defined to be a sequence of r pairs of input and output XOR differences each of which corresponds to one round. Differential cryptanalysis of an R-round block cipher relies on the existence of highly probable $(R-1)$-round characteristics. Whether this kind of characteristic exists in a block cipher depends heavily on the properties of the S-boxes. S-boxes that have a significantly non-uniform distribution of S-box difference pairs would help the occurrence of highly probable characteristics, where an S-box XOR difference pair refers to an input XOR and the corresponding output XOR of the S-box.

Differential cryptanalysis of a DES-like cipher which uses XOR keying is applied in this way: according to the structure of the cipher, the input to the last round and the input XOR values of the S-boxes in the last round can be directly obtained from the ciphertext. The output XOR values of some targeted S-boxes in the last round can be derived with a probability in the light of a characteristic. When a pair of the input and output differences of an S-box is known, the possible input values of the S-box are known. The possible sub-keys associated with each targeted S-box can consequently be achieved with a particular probability by using the knowledge of input to the last round and the possible input values to the S-box, where the sub-key of an S-box consists of those cipher key bits applied to the S-box. Precisely, with each plaintext pair, for each targeted S-box a set of bit-strings is achieved each member of which is a possible sub-key of the S-box, and among the set of these bit-strings, one bit-string is the correct sub-key of the S-box. By trying a number of plaintext pairs which have a specified XOR difference, a number of these kinds of sets of bit strings are obtained. Since a characteristic ideally occurs with a relatively high probability, the bit-string that is the correct sub-key of the

S-box should appear in these sets many more times than other bit-strings. The bit-string that appears in these sets most frequently is thus deduced as the correct sub-key of the S-box. One characteristic may involve a number of S-boxes in the last round and the key bits associated with these S-boxes can be deduced simultaneously using the characteristic. After using one or more characteristics, most of cipher key bits will be located. The rest of the cipher key bits may be easily located by using exhaustive key search.

Differential cryptanalysis of DES-like ciphers can be applied in a similar way to attack a basic SPN which uses XOR Keying. The difference between differential cryptanalysis of DES-like ciphers and that of a basic SPN is that differential cryptanalysis of a basic SPN locates the key bits XORed to the output instead of the key bits XORed to the input of the S-boxes in the last round. Also, differential cryptanalysis of a basic SPN uses the deterministic knowledge of the output of the last round and of the output XOR differences of the S-boxes in the last round, and the probabilistic knowledge of the input XOR differences of the S-boxes in the last round, to decide the sub-keys of the targeted S-boxes.

Heys and Tavares [12] found that linear transformations and S-boxes with good diffusion properties lower the probablility of the best probable $(R-1)$-round characteristic of an R-round SPN and increase resistance to differential cryptanalysis.

2.3.2 Linear Cryptanalysis

Matsui [16] presented the method of linear cryptanalysis to attack DES. The attack determines key bits by using a probable linear approximation of plaintext, ciphertext, and key bits which are likely to be satisfied. This is a known plaintext attack.

In the linear cryptanalysis of DES, the probable linear approximations of S-boxes are first investigated. Once a best probable linear approximation of an S-box is known, a best probable linear expression of a round can be obtained by using the knowledge of the round function. A probable linear expression of a round consists of the input, output, and sub-key bits of the round. After applying the best probable linear approximations of one round to a number of rounds and combining them, a probable cipher linear approximation can be achieved which consists of only plaintext, ciphertext, and key bits. Linear cryptanalysis then uses this kind of

probable linear approximation to locate the key bits using a hypothesis testing approach.

O'Connor [5] stated that as the size of an S-box gets larger the best probable linear approximation of the S-box tends to be worse and a cipher is more resistant to linear cryptanalysis. According to Nyberg's definition [19], the nonlinearity of an $n \times n$ bijective S-box is defined as the minimum nonlinearity of all non-zero linear combinations of output functions:

$$NL(S) = \min_{W_1,\ldots,W_n \in \{0,1\}, \text{all} W_i \neq 0} NL(\bigoplus_{i=1}^{n} W_i f_i) \tag{2.2}$$

where f_i represents the n-input function of the i-th output of the S-box, and the nonlinearity of an n-input boolean function, $f : \{0,1\}^n \rightarrow \{0,1\}$, is defined as the Hamming distance to the nearest affine function:

$$NL(f) = \min_{U_1,\ldots,U_n,V \in \{0,1\}} \#\{X | f(X) \neq \bigoplus_{i=1}^{n} U_i x_i \oplus V\} \tag{2.3}$$

where X is an n-bit vector, i.e. $X = [x_1, x_2, \ldots, x_n]$ with $x_i \in \{0,1\}$, $1 \leq i \leq n$.

Linear cryptanalysis can also be used to attack an SPN. To improve an SPN against linear cryptanalysis, Heys and Tavares [12] claimed that (i) large, randomly selected S-boxes are very likely to have high nonlinearity and (ii) the use of an appropriate linear transformation between rounds for the S-box interconnections increases the resistance to linear cryptanalysis.

Chapter 3

8×8 S-box Design

In this chapter, a new nonlinearity criterion for the design of 8×8 S-boxes is proposed. S-boxes satisfying this criterion and the diffusion criterion improve remarkably the ability of an SPN to resist linear cryptanalysis and differential cryptanalysis.

3.1 Introduction

A basic substitution-permutation encryption network consisting of a number of rounds of S-boxes connected by bit permutations is a straightforward implementation of a private-key block cipher [9]. The SPN structure is directly based on the concepts of "confusion " and "diffusion" introduced by Shannon [24]. Letting N represent the block size of a basic SPN composed of R rounds of $n \times n$ S-boxes, a simple example of an SPN with $N = 16$, $n = 4$, and $R = 3$ is illustrated in Figure 3.1. We assume that keying the network is realized by XORing the key bits with the data bits before each round of substitution and after the last round. The key bits associated with each round are derived from the cipher key according to some key scheduling algorithm. In this chapter we will consider an SPN with a block size of $N = 64$ using 8×8 S-boxes and whose interconnections are permutations where no two outputs of an S-box go to the same S-box in next round.

Like other block ciphers, an SPN is susceptible to linear cryptanalysis and differential cryptanalysis. Linear cryptanalysis suggested by Matsui [16] is a known-plaintext attack which uses some plaintext ciphertext pairs to break the cipher. Differential cryptanalysis, introduced by Biham and Shamir [3], is a chosen plaintext attack which examines the changes in ciphertext

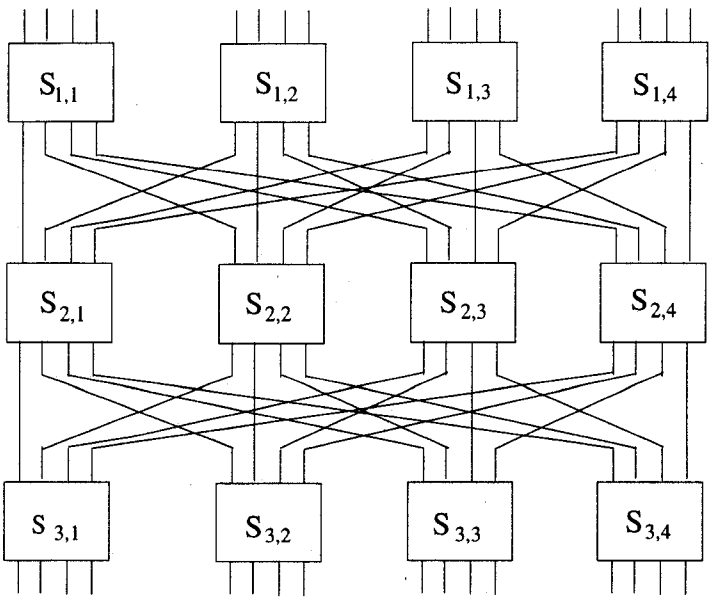


Figure 3.1: SPN with $N = 16, n = 4$, and $R = 3$

in response to controlled changes in the input. To attack an SPN, linear cryptanalysis is done by using knowledge of a highly probable linear approximation, while differential cryptanalysis is implemented by finding a highly probable differential characteristic. Both a highly probable linear approximation and a highly probable characteristic are achieved by exploiting the local properties of the network, specifically the S-box properties. Thus the design of S-boxes is crucial to the strength of an SPN.

In this chapter, a new S-box design criterion is proposed. S-boxes satisfying this criterion and the diffusion criterion [12] improve the ability of an SPN to resist linear cryptanalysis and differential cryptanalysis.

3.2 Background

In the application of linear cryptanalysis to SPN, the best r-round linear approximation of the form

$$P_{i_1} \oplus P_{i_2} \oplus \cdots \oplus P_{i_\alpha} \oplus C_{j_1} \oplus C_{j_2} \oplus \cdots \oplus C_{j_\beta} = K_{k_1} \oplus K_{k_2} \oplus \cdots \oplus K_{k_\gamma} \tag{3.1}$$

is of interest. This linear approximation is derived by combining a number of probable linear expressions of S-boxes from different rounds such that any intermediate terms (i.e., terms that are not plaintext, ciphertext, or key terms) are eliminated. The linear approximation of each

round which holds with some probability is obtained by using the knowledge of the linear approximations of S-boxes. As we will see later, when the linear approximations of S-boxes hold with small value of $|p_i - 1/2|$, where p_i denotes the probability of a linear approximation of an S-box, the overall cipher linear approximation will be satisfied with a small value of $|p_L - 1/2|$, which is inversely proportional to the number of plaintexts required to attack an SPN using a basic linear attack, where p_L represents the probability of the overall cipher linear approximation. Thus the nonlinearty property of an S-box needs to be considered in the context of linear cryptanalysis.

Definition 3.1 [16]: For a given $n \times n$ S-box, S, $NS(\alpha, \beta)$ is defined as the number of inputs to S, where a mod-2 linear combination of the input bits specified by vector α is equal to a mod-2 linear combination of output bits specified by vector β. In particular,

$$NS(\alpha, \beta) = \#\{x | x \in \{0,1\}^n, (\bigoplus_{i=0}^{n-1} \alpha[i] \cdot x[i] = \bigoplus_{i=0}^{n-1} \beta[i] \cdot s(x))\} \tag{3.2}$$

where the symbol $x[i]$ represents the i-th bit of vector x, $s(x)$ is the output of the S-box corresponding to input x, and $\oplus_{i=0}^{n-1} \alpha[i] \cdot x[i] = \oplus_{i=0}^{n-1} \beta[i] \cdot s(x)$ is referred to as a linear approximation of an S-box.

For some $\{\alpha, \beta\}$, the probability p of a linear approximation of an S-box is defined as $p = NS(\alpha, \beta)/2^n$. When $|p - 1/2|$ is small, then the linear expression defined by $\{\alpha, \beta\}$ is a poor approximation of the S-box.

For an R-round SPN, differential cryptanalysis is dependent on the existence of a highly probable $(R-1)$-round characteristic. An r-round characteristic is a list of difference pairs $\{(\Delta U_1, \Delta V_1), \ldots, (\Delta U_r, \Delta V_r)\}$, where $(\Delta U_i, \Delta V_i)$ represents the input XOR value and output XOR value in round i respectively, and $\Delta U_i = \Delta V_{i-1}$, $2 \leq i \leq r$, and ΔU_1 is the plaintext XOR difference. The existence of a highly probable $(R-1)$-round characteristic is determined by two factors [12]: (1) the distribution of an S-box XOR difference pairs $(\Delta x, \Delta y)$, where Δx is the bit-wise input XOR difference of 2 input vectors, x_1 and x_2, (i.e., $\Delta x = x_1 \oplus x_2$), and Δy is output XOR difference of an S-box, (i.e., $\Delta y = s(x_1) \oplus s(x_2)$), and (2) the diffusion of bit changes in the network.

Definition 3.2 [12]: An S-box satisfies a diffusion order of λ, $\lambda \geq 0$, if, for $wt\,(\Delta x) > 0$,

$$wt(\Delta y) > \begin{cases} \lambda + 1 - wt(\Delta x) & , wt(\Delta x) < \lambda + 1 \\ 0 & , wt(\Delta x) \geq \lambda + 1 \end{cases} \tag{3.3}$$

where Δx and Δy denote the input XOR difference and the corresponding output XOR difference of an S-box respectively, and $wt\,(\cdot)$ refers to the *Hamming weight* of the specified argument.

The diffusion order of an S-box is used to measure how many output bits can be changed while some of the input bits are changed, and it requires that the total number of changed input and output bits exceeds some assigned bound.

3.3 S-box Design Constraints

In this section, constraints on S-boxes that are good at strengthening an SPN against linear cryptanalysis and differential cryptanalysis are proposed.

3.3.1 Nonlinearity Requirement

In the linear cryptanalysis of an SPN, a cryptanalyst is interested in finding a linear approximation which is deduced by combining a number of probable linear approximations of the involved S-boxes. Suppose there are δ S-boxes involved in the derivation of a linear approximation of the overall cipher, and the probable linear expression of the i-th S-box holds with probability p_i. Also suppose that the key bits XORed to the data bits prior to entering the S-boxes are independent and uniformly random and consequently the inputs to the S-boxes involved in the derivation of the cipher linear approximation are independent and uniformly random. Then according to Lemma 3 in [16] the cipher linear approximation holds with probability

$$p_L = 1/2 + 2^{\delta-1} \prod_{i=1}^{\delta} (p_i - 1/2). \tag{3.4}$$

Also, it is shown in [16] that for a basic linear attack (algorithm 1) the number of known plaintexts required to guess the equivalent of a key bit (i.e. the right side of equation (3.1)) is approximately N_L, where

$$N_L = |p_L - 1/2|^{-2}. \tag{3.5}$$

By rewriting (3.4) as

$$p_L = 1/2 + 1/2 \prod_{i=1}^{\delta} (2p_i - 1), \tag{3.6}$$

it is evident that p_L, the probability of a linear approximation, is decided by two parameters: (1) δ, the number of S-boxes involved in the derivation of the linear approximation, and (2) p_i, the probability with which the linear expression of the i-th S-box is satisfied.

In [12], $|p_L - 1/2|$ was bounded by considering δ and $|p_i - 1/2|$ separately, without noticing that indirectly there exists a relationship beween δ and p_i. That is, when the probability of a probable linear expression of an S-box was bounded, no matter how many input and output bits are include in the linear expression, the probability p_i was bounded with the same relation

$$|p_i - 1/2| \leq |p_\varepsilon - 1/2| \tag{3.7}$$

where p_ε represents the probability of the best linear approximation and it can be described by

$$|p_\varepsilon - 1/2| = |(2^{n-1} - NL_{min})/2^n| \tag{3.8}$$

where n is the size of an S-box and NL_{min} is the lowest nonlinearity of any S-box in the cipher, i.e., for all S-boxes

$$|2^{n-1} - NS(\alpha, \beta)| \leq |2^{n-1} - NL_{min}|. \tag{3.9}$$

For a basic SPN, the permutation between two layers of S-boxes is usually arranged in this way: no two input bits of an S-box come from the same S-box in the previous round. Hence, intuitively, when a linear expression of an S-box includes more input and output bits, more S-boxes in the previous round and the next round will be caused to be involved in the derivation of a linear approximation of the overall cipher.

After studying the nature of the S-box interconnections in a basic SPN, it is not hard to see that, if a linear approximation of the overall cipher is obtained by combining the S-box linear approximations which include the same number of input plus output bits, the average number of S-boxes per round involved in the derivation of the linear approximation of the cipher is proportional to the number of input plus output bits which are included in each S-box linear approximation. Mathematically, let γ, $2 \leq \gamma \leq 2n$, represent the number of input plus output bits. Then, when γ-term S-box linear approximations are used to deduce a linear approximation

of an overall cipher, the per round number of S-boxes for deriving the linear approximation is at least $\gamma/2$.

Lemma 3.1 *Let γ, $2 \leq \gamma \leq 2n$, represent the number of input bits plus output bits in a linear approximation of an S-box. Assume no two input bits of an S-box come from the same S-box in the previous round, and all S-boxes use a γ-term linear approximation of a specific γ to derive the best possible cipher linear approximation. Then the best possible cipher linear approximation must involve $\gamma/2$ S-boxes on average per round.*

Proof: Consider an S-box in the r-th round. Since each input bit or output bit of an S-box connects to a different S-box in the previous or next round, based on assumption, the number of involved S-boxes in the previous and the next round must be at least γ.

Since we are considering a possible best cipher linear approximation, if a scenario in which the number of involved S-boxes in the previous and next round is γ exists, then the theorem is proven.

Figure 3.2 shows one of this kind of scenario for $n = 8$ and $\gamma = 4$. In this figure, the highlighted short lines represent the applied bits of S-boxes involved in the derivation of a cipher linear approximation. Since four bits of each involved S-box are used for the derivation of the cipher linear approximation, 4-term S-box linear approximations are used in this scenario. Obviously, the per round number of S-boxes involved in the cipher linear approximation is 2. Similarly, it can easily be verified that a linear approximation involving $\gamma/2$ S-boxes per round can be constructed for any value of γ. □

Now for a given value of γ, bound the linear approximation probability of an S-box by

$$\eta_i(\gamma) \leq \eta(\gamma) \tag{3.10}$$

where $\eta_i(\gamma) = |p_i(\gamma) - 1/2|$ with $p_i(\gamma)$ representing the probability of the γ-term linear expression for the i-th S-box of the linear approximation and $\eta(\gamma) = |p_\varepsilon(\gamma) - 1/2|$ with $p_\varepsilon(\gamma)$ representing the probability of the best γ-term linear expression of any S-box in the network.

Subsequently, based on Theorem 3.1 we have

$$\eta_L = 2^{\delta-1} \prod_{i=1}^{\delta} \eta_i(\gamma) \leq 2^{\delta-1} \prod_{i=1}^{\delta} \eta(\gamma) = 1/2(2\eta(\gamma))^{\gamma/2 \cdot R}, \tag{3.11}$$

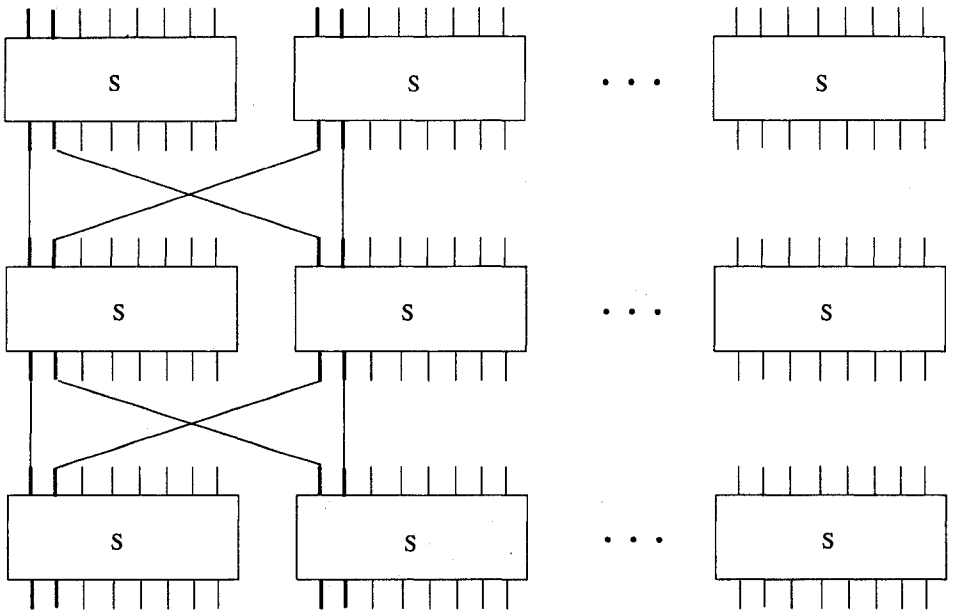


Figure 3.2: One possible best linear path when using 4-term linear approximations

where $\eta_L = |p_L - 1/2|$ and R is the number of rounds in the SPN. According to (3.11), to prevent a cryptanalyst from using some specific γ-term linear expressions to obtain a linear approximation with a higher probability, one could establish the relationship

$$(2\eta(\gamma_1))^{\gamma_1} = (2\eta(\gamma_2))^{\gamma_2} \tag{3.12}$$

for all $2 \leq \gamma_1, \gamma_2 \leq 2n$. Constraint (3.12) embodies the trade-off between the δ and p_i factors in deriving p_L and, assuming that a suitably small value for $\eta(2)$ can be found, should lead to a smaller value of p_L, than condition (3.7). However, to get the bound for the probability of the best linear approximation as small as possible, only putting constraint (3.12) on an S-box is not enough. In the context of an 8-round SPN consisting of 8×8 S-boxes, when a constraint similar to (3.12) is adopted to select S-boxes, using the relation

$$2\eta_i(\gamma) \leq \begin{cases} (2\eta(2))^{2/\gamma} & , 2 \leq \gamma \leq 3 \\ (2\eta(2))^{2/3} & , 4 \leq \gamma \leq 2n \end{cases} \tag{3.13}$$

as the constraint put on S-boxes leads to the tightest bound on the probability of the best linear approximation, as the following discussion shows.

In our experimentations for selecting S-boxes, with either the constraint (3.12) or

$$2\eta_i(\gamma) \leq \begin{cases} (2\eta(2))^{2/\gamma} & , 2 \leq \gamma \leq 4 \\ (2\eta(2))^{2/4} & , 5 \leq \gamma \leq 2n \end{cases} \tag{3.14}$$

the minimum $2\eta(2)$ achieved is the same value of $|2 \times \frac{114}{256} - 1| = 7/64$. In the experiment, we use random numbers to generate the outputs of an S-box. Every time a random number is generated, it is checked whether this number can be the value of an S-box output that corresponds to the current S-box input so that all of the S-box output values selected so far satisfy our design citerion. Since the minimum $2\eta(2)$ achieved is the same value of $7/64$, the probability of the best possible cipher linear approximation which is derived from an SPN consisting of S-boxes satisfying (3.14) must not be greater than that of the possible best cipher linear approximation which is derived from an SPN consisting of S-boxes satisfying (3.12). In other words, (3.14) sets a tighter bound on the probability of the best linear approximation than (3.12).

To compare the constraint of (3.13) with that of (3.14), the probability of the best possible linear approximation of cipher needs to be calculated. When the constraint (3.13) is put on S-boxes, the minimum $2\eta(2)$ achieved through experiment was $|2 \times \frac{112}{256} - 1| = 1/8$. For an 8-round SPN, under condition (3.13) and (3.14) the equivalent number of 2-term S-box linear approximations for deriving a best linear approximation are 7.333 and 6.8333, respectively, where equivalent number of 2-term linear approximation means that a best γ-term S-box linear approximation is equivalent to $2/\gamma$ best 2-term S-box linear approximation. Hence, under condition (3.13) the best probability is $|2 \times \frac{112}{256} - 1|^{7.333} = (\frac{1}{8})^{7.333} = 2.386 \times 10^{-7}$, and under condition (3.14) it is $|2 \times \frac{114}{256} - 1|^{6.833} = (\frac{7}{64})^{6.833} = 2.707 \times 10^{-7}$. Since $2.386 \times 10^{-7} < 2.707 \times 10^{-7}$, the constraint (3.13) can be used to set a tighter bound on the probability of the best cipher linear approximation than (3.14).

Noticing that constraint (3.12) is the loosest among the similar constraints and the minimum $2\eta(2)$ achievable is $|\frac{7}{64}|$, we have actually considered all the possible best linear approximations under all similar constraints. Therefore we conclude that relation (3.13) provides the tightest bound on the probability of the best linear approximation.

In Figure 3.3 we give the algorithm for an 8-round cipher to compute the equivalent number of 2-term S-box linear approximations involved in the best possible cipher linear approximation based on constraint (3.13). In this algorithm, the equivalent number of S-boxes involved in a cipher linear approximation of an R-round SPN is calculated, and only the best possible cipher

Initialize $k_{min} = R$;
For all $[n_1, n_2, \ldots, n_R] \in \{1, 2, \ldots, 8\}^R$ do
$k = \frac{1}{2}(n_1 + n_R)$
For $r = 2$ to $R - 1$ do
$n \leftarrow n_{r-1} + n_{r+1}$
if $(n > 4)$ $n = 4$
$k = 2n_r/n + k$
if $k < k_{min}$ then $k_{min} = k$
output: k_{min}

Figure 3.3: Algorithm for computing the equivalent number of 2-term S-boxes

linear approximations are taken into account, i.e., assuming any involved S-box in the $(r-1)$-th or $(r+1)$-th round offers a bit to any involved S-box in the r-th round. During the calculation, we use n_r to denote the number of S-boxes in the r-th round which are involved in a linear approximation of the overall cipher, and $k = 1/2(n_1 + n_R)$ is the equivalent number of S-boxes in the first and last round.

According to algorithm of Figure 3.3, for an 8-round SPN, with constraint (3.14) the smallest equivalent number of 2-term S-boxes involved in a linear approximation is 6.833. For example, one scenario is that from the 1st to 8-th round, the number of S-boxes is 1, 1, 1, 2, 3, 2, 1, and 1, respectively.

Let us calculate the number of known plaintexts required in the basic linear attack. As mentioned above, under condition (3.13) the equivalent number of 2-term linear expressions involved in the best linear approximation of an 8-round SPN is 22/3 (i.e., 7.333). Therefore according to (3.11), for an 8-round SPN the greatest value of $|p_L - 1/2|$ is $1/2(2\eta(2))^{22/3}$. This signifies that in the basic linear attack the number of plaintexts required to deduce one equivalent bit of key is at least $4/(2\eta(2))^{44/3}$. From the results of our experiments, 8×8 S-boxes satisfying (3.13) with $2\eta(2) = 1/8$ can be achieved. Hence, if an 8-round SPN is constructed using 8×8 S-boxes satisfying (3.13) with $2\eta(2) = 1/8$, it requires at least 2^{46} known plaintexts to deduce one equivalent key bit using the basic linear attack.

In contrast, in [12], (3.7) is used to bound the probability of a linear expression of an S-box, and the value of the minimum $2\eta(2)$ is assumed to be $|2 \times 96/256 - 1| = 1/4$. Since the minimum number of S-boxes involved in a cipher linear approximation is 8, the resulting $|p_L - 1/2|$ is

	8 × 8 S-box			
λ	$\eta(2)$ under condition (3.14)	%	$\eta(2)$ under condition (3.13)	%
0	7/128	2.65	1/16	0.004
1	7/128	3.90	1/16	0.267

Table 3.1: Proportion of S-boxes satisfying nonlinearity requirement selected from S-boxes having diffusion order λ

$1/2(2\eta(2))^8 = 2^{17}$ and the number of required plaintexts in a basic linear attack could be as few as 2^{34}.

3.3.2 Diffusion Order Requirement

S-boxes with a high diffusion order can enhance the ability of an SPN to resist differential cryptanalysis [3]. Letting N represent the block size of an SPN, n represent the size of an S-box, M represent the number of S-boxes used in each round where $M = N/n$, and Π represent the set of permutations for which no two outputs of an S-box are connected to one S-box in the next round, according to Theorem 1 in [12], for an R-round SPN which uses a permutation $\pi \in \Pi$, where R is a multiple of 4 and $M \geq n$, the probability of an $(R-1)$-round characteristic satisfies

$$p_{\Omega_{R-1}} \leq (p_\delta)^{\frac{\lambda+2}{2}R-(\lambda+1)} \tag{3.15}$$

where all S-boxes satisfy diffusion order λ and p_δ represents the maximum S-box XOR pair probability. The maximum λ found in [12] for 8×8 S-boxes is $\lambda = 2$.

By using the *depth-first-search* algorithm in [12], 8×8 S-boxes are examined for their nonlinearity property and diffusion order. It is determined that it is not difficult to find S-boxes with diffusion order of 1 which satisfy the suggested nonlinearity requirement with a small value of $2\eta(2)$. Some results are illustrated in Table 3.1. It is clear that S-boxes which satisfy the nonlinearity requirement of either (3.13) or (3.14) and at the same time have a good diffusion property can be found.

3.4 Conclusion

Based on the observation that the number of S-boxes for deriving an overall linear approximation is on average related to the number of input plus output bits of each involved S-box linear

approximation, the restriction on the probability of an S-box linear approximation with more input plus output bits is relaxed. Thus S-boxes whose fewer-term linear approximations are poorer are found. As a result, the ability to resist linear cryptanalysis of an SPN that is constructed from these S-boxes is improved remarkably.

To thwart differential cryptanalysis, it is desirable that S-boxes satisfy a relatively high diffusion order. S-boxes that satisfy our suggested nonlinearity requirement can be selected from the S-boxes with diffusion order of 1. Thus, the new S-box design criterion suggested in this chapter would help us to obtain good S-boxes which enable an SPN to be stronger in resisting linear cryptanalysis while still having properties suitable for resisting differential cryptanalysis.

Chapter 4

Linear Transformation Design

In this chapter, a new linear transformation is proposed as the method of interconnection between rounds of S-boxes. It is shown that when the linear transformation is adopted in an SPN, the average number of equivalent 2-term S-boxes per round involved in a cipher linear approximation has a lower bound of 1.5.

4.1 Rearranging Permutations

As was mentioned in the previous chapter, a substitution-permutation network is vulnerable to the attack of linear cryptanalysis. In a basic SPN, any two rounds of S-boxes are connected by a bit permutation such that no two input bits of an S-box come from the same S-box in the previous round. To improve an SPN against linear cryptanalysis, one possible method might be to rearrange the permutation for each round, according to the probabilities of all of the linear approximations derived up to the previous round.

To test the feasibility of the method, we only considered the approximations that are derived using only 2-term linear approximations and assumed that the S-boxes to which the output bits in the $(r-1)$-th round are connected in the r-th round are decided before arranging the permutation of the bits within each S-box but which bit of these output bits in the $(r-1)$-th round will connect to which bit of the input bits of the S-box is to be decided. The connections between these output bits in the $(r-1)$-th round and these input bits of the S-box are decided in this way: if an output bit is included in the linear approximation which is the k-th best among the linear approximations each of which includes one of the output bits in the $(r-1)$-th

round, it is connected to the input bit which is included in the S-box linear approximation which is the k-th worst among all of the 2-term S-box linear approximations of the S-box. For example, suppose bits $i_1, i_2, \ldots, i_8$ in the $(r-1)$-round are connected to the bits $j_1, j_2, \ldots, j_8$ of an 8×8 S-box in the next round, it is for bit i_1 that the relatively best linear approximation exists, and it is for input bit j_1 that the relatively worst 2-term linear approximation exists. Then bit i_1 is connected to bit j_1.

However, the results of our simulation experiments demonstrate that this method is not effective. For instance, in an SPN which is constructed with 8×8 S-boxes satisfying the nonlinearity criterion

$$2\eta_i(\gamma) \leq \begin{cases} (2(\eta(2))^{2/\gamma} & ,2 \leq \gamma \leq 4 \\ (2\eta(2))^{2/4} & ,5 \leq \gamma \leq 2n \end{cases} \tag{4.1}$$

proposed in Chapter 3, the following result was derived: after designing the bit permutation, the value of $|2p_L - 1|$ of the best cipher linear approximation is 1.697×10^{-8}; while in theory $|2p_L - 1|$ is bounded by $|2p_L - 1| \leq (7/64)^{6.833} = 2.7073 \times 10^{-7}$ in Chapter 3, where p_L denotes the probability of a cipher linear approximation as in Chapter 3. It should also be noted that a problem with this approach is that a changing permutation is hard to implement. When a changing permutation is implemented, the permutations in different rounds may be different from each other, so for an R-round SPN, $(R-1)$ different permutations need to be implemented. For the VLSI implementeation of an SPN, certainly this kind of implemenation is hard to do, because too many wirings have to be arranged.

To strengthen an SPN against linear attack without augmenting the number of rounds, an effective way is to use a linear transformation other than a permutation between rounds of S-boxes. In this chapter we introduce a linear transformation that is intended to be used in an SPN that contains 16 4×4 S-boxes in each round. The use of the linear transformation results in the average number of equivalent 2-term S-boxes per round involved in a cipher linear approximation having a lower bound of 1.5 which is an improvement over straightforward lower bound of 1.

4.2 Motivation for the use of 4×4 S-boxes

As the linear transformation is designed to be used in an SPN consisting of 4×4 S-boxes, before the linear transformation is discussed, we would like to give the motivation for designing a linear transformation that is to be used in an SPN consisting of 4×4 S-boxes.

Many cryptographic applications require a single chip implementation of a cryptographic algorithm. To make it as easy as possible for an LSI logic designer to realize the design on a single chip, the minimum number of logic gates required to implement an SPN needs to be small. Decreasing the minimum number of logic gates in an SPN motivated us to study SPNs consisting of small S-boxes. The reason that small S-boxes can reduce the number of logic gates is given below.

In hardware, an S-box is usually implemented as a set of boolean functions instead of a table lookup. An $n \times n$ S-box is a function that maps n input bits $(x_1, x_2, \ldots x_n)$ onto n output bits $(y_1, y_2, \ldots, y_n)$. Each output bit, y_i, can be represented as a Boolean expression of the input bits, i.e., y_i can be represented as one or more minterms that are combined using a logic OR operation, where a minterm includes all of input bits or their complements which are ANDed together.

For example, if y_i can be expressed in terms of inputs (x_1, x_2, x_3) as follows:

x_1	x_2	x_3	y_1	y_2	y_3
0	0	0	1	1	0
1	0	0	0	1	1
0	1	0	0	0	1
1	1	0	1	0	0
0	0	1	0	1	1
1	0	1	1	0	1
0	1	1	1	0	1
1	1	1	0	1	0

then the Boolean expression for y_i is $y_1 = \overline{x_1}x_2\overline{x_3} + x_1x_2\overline{x_3} + x_1\overline{x_2}x_3 + \overline{x_1}x_2x_3$, $y_2 = \overline{x_1}x_2\overline{x_3} + x_1\overline{x_2}\overline{x_3} + \overline{x_1}\overline{x_2}x_3 + x_1x_2x_3$, and $y_3 = x_1\overline{x_2}\overline{x_3} + \overline{x_1}x_2\overline{x_3} + \overline{x_1}\overline{x_2}x_3 + x_1\overline{x_2}x_3 + \overline{x_1}x_2x_3$, where "+" denotes the logical OR operation.

It is easy to find that, for an 8×8 S-box, the Boolean expression for an output bit y_i may include at most 2^8 8-bit minterms, but for 4×4 S-box, an output bit y_i can include at most

2^4 4-bit minterms. Also, considering the cryptographic properties of S-boxes,it is usually true that the bigger the size of an S-box the more the minterms included in the Boolean expression for an output bit. Therefore it is important to use small S-boxes to reduce the number of logic gates needed in the implementation of an SPN.

In the process of exploiting the properties of small S-boxes, 3×3 and 4×4 S-boxes were both studied. After searching all 3×3 S-boxes, we found that the minimum $2\eta(2)$ that a 3×3 S-box can reach is 0.5. Since the value of 0.5 for $2\eta(2)$ appears to introduce a very high requirement on the number of rounds or the linear transformation of an SPN, we feel that 3×3 S-boxes are not suitable to be adopted for use in an SPN.

However, for 4×4 S-boxes, the situation is different. 4×4 S-boxes that satisfy diffusion order of 1 and meanwhile satisfy our new nonlinear criterion proposed in Chapter 3 with $2\eta(2) = 1/4$ can be found. This property of 4×4 S-boxes provides us with the confidence that 4×4 S-boxes are promising candidates for efficient, secure SPNs. Thus we have studied the structure of an SPN that are constructed from 4×4 S-boxes and have specifically focused on developing an appropriate linear transformation to be used to efficiently interconnect rounds of S-boxes.

4.3 Equivalent Number of 2-term S-boxes

The concept of an equivalent 2-term S-box is critical to our description, so we should begin with a careful analysis of this concept.

In our substitution-permutation network, the S-boxes are selected such that they satisfy two criteria : 1) For the differential property, the diffusion order of an S-box is 1; 2) For the nonlinear property, $2\eta_i(\gamma)$ satisfies

$$2\eta_i(\gamma) \leq \begin{cases} |2 \times \frac{6}{16} - 1| = 1/4; & \gamma = 2 \\ |2 \times \frac{4}{16} - 1| = 1/2; & 3 \leq \gamma \leq 2n \end{cases} \tag{4.2}$$

where γ is the number of input plus output bits included in a linear approximation of an S-box. Note that (4.2) is similar to (3.12). Although it is looser, it is selected so that $\eta(2)$ can be small. How to select these S-boxes is discussed in the previous chapter.

By rule (4.2) and the trivial relation of $(1/2)^2 = 1/4$, we define that:

• If an S-box for which $\gamma = 2$ is involved in a linear approximation, then for this S-box the equivalent number of 2-term S-boxes is 1;

• If an S-box for which $\gamma \geq 3$ is used in a linear approximation, then for this S-box the equivalent number of 2-term S-boxes is 0.5.

In this way, we shall use the equivalent number of 2-term S-boxes involved in a best linear approximation to measure the level of the security of an SPN in thwarting linear cryptanalysis. The greater the total number of equivalent 2-term S-boxes involved in the best linear approximation, the stronger an SPN is in resisting linear cryptanalysis.

4.4 A Review of Previously Proposed Linear Transformations

A permutation can be viewed as a special kind of linear transformation. The use of a permutation in an SPN is not able to eliminate the cases where in each round only one 2-term S-box is involved in a linear approximation. In other words, in each round only one 2-term linear expression of an S-box is involved in a linear approximation. Thus, for an SPN that employs permutations between rounds of S-boxes, to set a lower bound for the equivalent number of 2-term S-boxes involved in a linear approximation, the bound should never be greater than 1.

The use of other linear transformations to interconnect rounds of S-boxes can increase the number of equivalent 2-term S-boxes involved in a linear approximation. The use of linear transformation in an SPN is first suggested by the authors in [12], in which the authors designed a linear transformation that makes at least 1.5 S-boxes in each round be involved in a linear approximation. Later, the authors in [32] proposed another linear transformation, with which the average per round number of S-boxes involved in a linear approximation is 2. Both of these two linear transformations strongly improve an SPN's resistance to linear cryptanalysis. However, when an SPN is constructed from S-boxes that satisfy relation (4.2), these two linear transformations give no advantage.

The linear transformation proposed in [12] is defined by $V = \pi[L(U)]$, where $V = [V_1 , V_2, \ldots , V_N]$ is the vector of input bits of a round of S-boxes, $U = [U_1, U_2 , \ldots, U_N]$ is the vector of bits from the previous round output, π is a permutation for which no two outputs of an

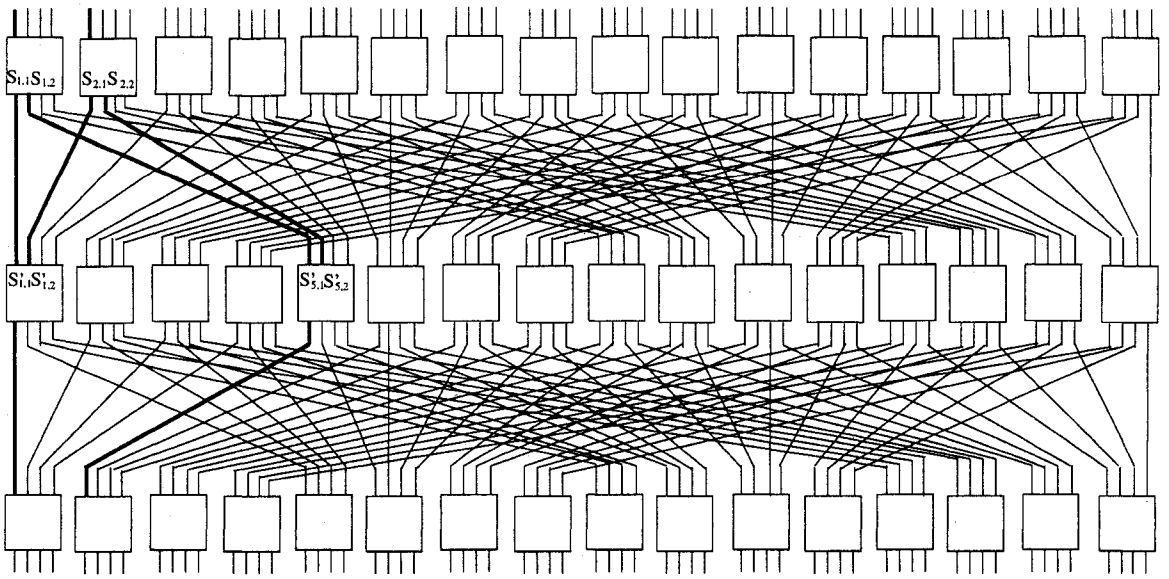


Figure 4.1: Ayoub's Permutation

S-box are connected to one S-box in the next round, and $L(u) = [L_1(U), \ldots, L_N(U)]$, where $L_i(U) = U_1 \oplus U_2 \oplus \ldots \oplus U_{i-1} \oplus U_{i+1} \oplus \ldots \oplus U_N$. When an SPN is combined with this linear transformation and the sort of 4×4 S-boxes that satisfy relation (4.2), the lower bound for the equivalent number of 2-term S-boxes can not exceed 1. This means it is possible that the per round number of equivalent 2-term S-boxes involved in a linear approximation can be 1.

We give an example to illustrate this conclusion. In the definition of the linear transformation, π is any kind of permutation for which no two output bits of an S-box are connected to one S-box in the next round. Hence Ayoub's permutation [2] satisfies this condition and can be used in the linear transformation. Ayoub's permutation for a 64-bit network is illustrated in Figure 4.1 and is described as follows: the output bits in the first bit positions of the 16 S-boxes in this round are successively connected to the input bits of the first 4 S-boxes in the next round, the output bits in the second bit positions of the 16 S-boxes in this round are successively connected to the input bits of the second 4 S-boxes in the next round, ..., and so on.

Suppose we use the highlighted linear path to achieve a linear approximation. In accordance with the definition of the linear transformation, we immediately have these relations: $S'_{1,1} = S_{1,1} \oplus Q$, $S'_{1,2} = S_{2,1} \oplus Q$, $S'_{5,1} = S_{1,2} \oplus Q$, and $S'_{5,2} = S_{2,2} \oplus Q$, where $Q = U_1 \oplus U_2 \oplus \ldots \oplus U_{N-1} \oplus U_N = S_{1,1} \oplus S_{1,2} \oplus S_{1,3} \oplus S_{1,4} \oplus \ldots \oplus S_{16,1} \oplus S_{16,2} \oplus S_{16,3} \oplus S_{16,4}$, with $S_{i,j}$ representing

the j-th output bit of the i-th S-box in the previous round and $S'_{i,j}$ representing the j-th input bit of the i-th S-box in the next round. Combining these expressions together, we have

$$S'_{1,1} \oplus S'_{1,2} \oplus S'_{5,1} \oplus S'_{5,2} = S_{1,1} \oplus S_{1,2} \oplus S_{2,1} \oplus S_{2,2}. \tag{4.3}$$

Also, according to the connections of Figure 4.1, we get

$$S'_{1,1} \oplus S'_{2,1} = S_{1,1} \oplus S_{5,1}. \tag{4.4}$$

By relations (4.3) and (4.4), it is clear that a linear approximation that involves S-boxes S_1, S_2, S'_1, S'_5 in every two rounds exists. Since for each of these 4 S-boxes, a 3-term linear expression is used, the per round number of equivalent 2-term S-boxes involved in the linear approximation is 1.

We now consider the linear transformation presented in [32] and described by

$$S'_{i,j} = \bigoplus_{k=1}^{m} S_{k,j} \oplus S_{i,j} \tag{4.5}$$

where m represents the number of S-boxes in one round of an SPN, $S'_{i,j}$ denotes the j-th input bit of i-th S-box in one round, and $S_{k,j}$ refers to the j-th output bit of k-th S-box in the previous round. For an SPN that consists of 4×4 S-boxes satisfying our design criteria, this linear transformation can only guarantee that the per round number of equivalent 2-term S-boxes involved in a linear approximation is not less than 1 as shown in the following lemma.

Lemma 4.1 *Under the linear transformation represented by (4.5), the per round number of equivalent 2-term S-boxes involved in a linear approximation can be 1.*

Proof. Without loss of generality, suppose a cryptanalyst adopts S-boxes $S1$ and $S2$ to find a linear path, as $S'_{1,1} = \oplus_{k=1}^{m} S_{k,1} \oplus S_{1,1}$, $S'_{2,1} = \oplus_{k=1}^{m} S_{k,1} \oplus S_{2,1}$, $S'_{1,2} = \oplus_{k=1}^{m} S_{k,2} \oplus S_{1,2}$, and $S'_{2,2} = \oplus_{k=1}^{m} S_{k,2} \oplus S_{2,2}$. It follows that $S'_{1,1} \oplus S'_{1,2} \oplus S'_{2,1} \oplus S'_{2,2} = S_{1,1} \oplus S_{1,2} \oplus S_{2,1} \oplus S_{2,2}$. Thus in every round each of S-boxes $S1$ and $S2$ contributes 4 bits to a linear path. As stated above, for an S-box that offers a 4-term linear expression to a linear path, its equivalent number of 2-term S-boxes is 0.5. Hence in this case the per round number of equivalent 2-term S-boxes is 1. □

We now propose a linear transformation which based on the S-box property of (4.2) can be used to guarantee a better bound on the average number of equivalent 2-term S-box approximations per round.

4.5 Linear Transformation Design

In this section, our linear transformation shall be described in detail. In order to make the description clear, it is important to define some notational conventions first.

From now on, we will use $S_{i,j}$ to denote the j-th output bit of i-th S-box in one round, and $S'_{i.j}$ to denote the j-th input bit of i-th S-box in the next round, where $1 \leq i, j \leq m$, and m is the number of S-boxes in one round. For example, while $S_{1,2}$ represents output bit 2 of S-box $S1$ in round r, $S'_{1,2}$ represents input bit 2 of S-box $S1$ in round $r+1$.

As well, we separate the 16 S-boxes in one round into 4 groups, each of which is a combination of 4 S-boxes, and we refer each group as a partition. Partition i includes 4 S-boxes Sk, where $k = 4(i-1)+j$, $1 \leq j \leq 4$. For example, partition 2 comprises of $S5, S6$, $S7$, and $S8$.

Moreover, the notation Q_k^{ij} is defined to denote the XOR value of the k-th bits of all the S-boxes in partition i and j. For example, Q_3^{41} represents the XOR of the 3rd bit of the S-boxes in partition 4 and in partition 1, i.e., $Q_3^{41} = S_{13,3} \oplus S_{14,3} \oplus S_{15,3} \oplus S_{16,3} \oplus S_{1,3} \oplus S_{2,3} \oplus S_{3,3} \oplus S_{4,3}$ $= \oplus_{i=13}^{16} S_{i,3} \oplus_{i=1}^{4} S_{i,3}$.

By now all the notational conventions have been discussed. The designed linear transformation is expressed in Table 4.1. In Table 4.1, $i \in \{1,2,3,4\}$, $j \in \{5,6,7,8\}$, $k \in \{9,10,11,12\}$, and $l \in \{13,14,15,16\}$. For example, $S'_{1,1} = Q_1^{12} \oplus S_{1,1} = \oplus_{i=1}^{8} S_{i,1} \oplus S_{1,1}$, and $S'_{11,2} = Q_2^{12} \oplus S_{(11-8),2}$ $= \oplus_{i=1}^{8} S_{i,2} \oplus S_{3,2}$,

In the proof of the theorem that under this linear transformation the per round number of equivalent 2-term S-boxes involved in a linear transformation is not less than 1.5, the inverse linear transformation is used. The inverse linear transformation can be trivially deduced from the original one and its expression is exactly the same as the original.

Lemma 4.2 *The inverse linear transformation is the same as original.*

Partition		1	2	3	4
S-boxes		$S_i^{'}$, $1 \leq i \leq 4$	$S_j^{'}$, $5 \leq j \leq 8$	$S_k^{'}$, $9 \leq k \leq 12$	$S_l^{'}$, $13 \leq l \leq 16$
Bit	1	$Q_1^{12} \oplus S_{i,1}$	$Q_1^{12} \oplus S_{j,1}$	$Q_1^{34} \oplus S_{k,1}$	$Q_1^{34} \oplus S_{l,1}$
	2	$Q_2^{34} \oplus S_{(i+8),2}$	$Q_2^{34} \oplus S_{(j+8),2}$	$Q_2^{12} \oplus S_{(k-8),2}$	$Q_2^{12} \oplus S_{(l-8),2}$
	3	$Q_3^{41} \oplus S_{(i+12),3}$	$Q_3^{23} \oplus S_{(j+4),3}$	$Q_3^{23} \oplus S_{(k-4),3}$	$Q_3^{41} \oplus S_{(l-12),3}$
	4	$Q_4^{23} \oplus S_{(i+4),4}$	$Q_4^{41} \oplus S_{(j-4),4}$	$Q_4^{41} \oplus S_{(k+4),4}$	$Q_4^{23} \oplus S_{(l-4),4}$

Table 4.1: Linear Transformation

Proof (Sketch): If we prove that the expression for $S_{i,j}$ is same as for $S_{i,j}^{'}$, the lemma is proven. Without loss of generality, we prove that the expression for $S_{1,3}^{'}$ is same as for $S_{1,3}$. A similar approach can be taken for all other values of i and j.

As defined in the linear transformation, $S_{1,3}^{'}$ is expressed by

$$S_{1,3}^{'} = Q_3^{41} \oplus S_{13,3} \tag{4.6}$$

where $Q_3^{41} = S_{13,3} \oplus S_{14,3} \oplus S_{15,3} \oplus S_{16,3} \oplus S_{1,3} \oplus S_{2,3} \oplus S_{3,3} \oplus S_{4,3}$.

Also, $S_{13,3}^{'}$, $S_{14,3}^{'}$, $S_{15,3}^{'}$, $S_{16,3}^{'}$, $S_{2,3}^{'}$, $S_{3,3}^{'}$, and $S_{4,3}^{'}$ are defined as: $S_{13,3}^{'} = Q_3^{41} \oplus S_{1,3}$, $S_{14,3}^{'} = Q_3^{41} \oplus S_{2,3}$, $S_{15,3}^{'} = Q_3^{41} \oplus S_{3,3}$, $S_{16,3}^{'} = Q_3^{41} \oplus S_{4,3}$, $S_{2,3}^{'} = Q_3^{41} \oplus S_{14,3}$, $S_{3,3}^{'} = Q_3^{41} \oplus S_{15,3}$, and $S_{4,3}^{'} = Q_3^{41} \oplus S_{16,3}$.

Thus,

$$\begin{aligned} & S_{13,3}^{'} \oplus S_{14,3}^{'} \oplus S_{15,3}^{'} \oplus S_{16,3}^{'} \oplus S_{1,3}^{'} \oplus S_{2,3}^{'} \oplus S_{3,3}^{'} \oplus S_{4,3}^{'} \\ = \; & Q_3^{41} \oplus S_{1,3} \oplus Q_3^{41} \oplus S_{2,3} \oplus Q_3^{41} \oplus S_{3,3} \oplus Q_3^{41} \oplus S_{4,3} \oplus \\ & Q_3^{41} \oplus S_{13,3} \oplus Q_3^{41} \oplus S_{14,3} \oplus Q_3^{41} \oplus S_{15,3} \oplus Q_3^{41} \oplus S_{16,3} \\ = \; & S_{13,3} \oplus S_{14,3} \oplus S_{15,3} \oplus S_{16,3} \oplus S_{1,3} \oplus S_{2,3} \oplus S_{3,3} \oplus S_{4,3} \\ = \; & Q_3^{41}. \end{aligned}$$

Now let us define

$$Q_3^{'41} = S_{13,3}^{'} \oplus S_{14,3}^{'} \oplus S_{15,3}^{'} \oplus S_{16,3}^{'} \oplus S_{1,3}^{'} \oplus S_{2,3}^{'} \oplus S_{3,3}^{'} \oplus S_{4,3}^{'}.$$

Then

$$\begin{aligned} Q_3^{'41} \oplus S_{13,3}^{'} &= Q_3^{41} \oplus S_{13,3}^{'} \\ &= Q_3^{41} \oplus Q_3^{41} \oplus S_{1,3} \\ &= S_{1,3} \end{aligned}$$

i.e.,

$$S_{1,3} = Q_3^{'41} \oplus S_{13,3}^{'}. \tag{4.7}$$

Comparing the form of Q_3^{41} with that of $Q_3'^{41}$, and $S_{1,3}$ with $S'_{1,3}$, we conclude that the expression for $S_{1,3}$ is exactly the same as for $S'_{1,3}$. □

We now consider developing a theorem which shows when an SPN adopts the linear transformation of Table 4.1 the average per round number of equivalent 2-term S-boxes involved in a cipher linear approximation is at least 1.5 (except for the first and last round). The theorem is going to be proven by showing that the following statements are true and then combining them:

(i) If only one S-box in one round is involved in an overall linear approximation, then in either the next round or previous round the number of S-boxes involved in the linear approximation must be at least 7;

(ii) If 2 S-boxes in one round are involved in the derivation of a linear approximation and at least one of them contributes a γ-term, $\gamma > 2$, linear expression to the linear approximation, then in the previous or next round there must be at least 4 S-boxes involved in the linear approximation;

(iii) If 2 S-boxes in one round are involved in the derivation of a linear approximation and each of them offers a 2-term linear expression to the linear approximation, then for this round the number of equivalent 2-term S-boxes is 2;

(iv) If 2 S-boxes in round r are involved in the derivation of a linear approximation such that 4 S-boxes in the previous round are involved, then each of the 4 S-boxes in round $r-1$ must only offer one output bit for the derivation of the linear approximation;

(v) If 3 or more S-boxes in one round are involved in the derivation of a linear approximation, then for this round the number of equivalent 2-term S-boxes is at least 1.5.

Statements (iii) and (v) are trivially obvious. So we should only focus on arguments for statements (i), (ii) and (iv).

Lemma 4.3 *If one S-box in one round is involved in the derivation of a linear approximation, then the number of involved S-boxes in the previous round must be at least 7.*

Proof. Without loss of generality, suppose S-box $S1$ in one round is involved in the derivation

of an overall linear approximation. Then the 4 input bits of the S-box are:

$$S'_{1,1} = Q_1^{12} \oplus S_{1,1},$$

$$S'_{1,2} = Q_2^{34} \oplus S_{(1+8),2},$$

$$S'_{1,3} = Q_3^{41} \oplus S_{(1+12),3},$$

and

$$S'_{1,4} = Q_4^{23} \oplus S_{(1+4),4}.$$

Since (1) each $S'_{1,i}$ $1 \leq i \leq 4$ consists of 7 bits from different 7 S-boxes and (2) when $i \neq j$, $1 \leq i, j \leq 4$, the bits included in $S'_{1,i}$ are different from the bits included in $S'_{1,j}$ (i.e., they can not cancel each other), no matter how many input bits of $S1$ are involved in the deduction of a linear approximation, at least 7 bits which come from 7 different S-boxes in the previous round will be included in the linear approximation. This means that at least 7 S-boxes in the previous round will be involved in a linear approximation. Similar results can be shown for any S-box $Sk, 2 \leq k \leq 16$. □

Statement (i) is now proven. We turn our attention to the proof of statement (ii). When several S-boxes in one round are involved in a linear approximation, some of their input and output bits will take part in the derivation of the linear approximation. On these involved bits, we have Lemma 4.4 and 4.5.

Lemma 4.4 *Among the involved input bits in one round, if there exists one bit such that its bit position is different from that of all of the other bits, then the number of involved S-boxes in the previous round must be at least 7.*

Proof. Without loss of generality, suppose $S'_{1,1}$ is used in the derivation of a linear approximation, but none of $S'_{i,1}$, $2 \leq i \leq 16$, are.

By the definition of the linear transformation,

$$\begin{aligned} S'_{1,1} &= Q_1^{12} \oplus S_{1,1} \\ &= S_{2,1} \oplus S_{3,1} \oplus S_{4,1} \oplus S_{5,1} \oplus S_{6,1} \oplus S_{7,1} \oplus S_{8,1}. \end{aligned}$$

Since none of $S'_{2,1}$, $S'_{3,1}$ $\ldots$, $S'_{16,1}$ is involved in the derivation of the linear approximation, any one of $S_{2,1}$, $S_{3,1}$, $S_{4,1}$, $S_{5,1}$, $S_{6,1}$, $S_{7,1}$, $S_{8,1}$ which constitutes $S'_{1,1}$ can not be cancelled during the derivation of the linear approximation. This implies that 7 S-boxes $S2$, $S3$, $S4$, $S5$, $S6$, $S7$, and $S8$ in the previous round must be involved in the linear approximation. □

Lemma 4.5 *If 2 S-boxes in one round are involved in a linear approximation, then in the previous round the possible number of involved S-boxes is 2, 4, 6, or greater than or equal to 7.*

Proof. The proof of the lemma is done by considering 2 cases: Case 1: 2 involved S-boxes in one round are in the same partition; and Case 2: 2 involved S-boxes belong to 2 different partitions. For case 2, there are 6 subcases to be considered:

Case 2.1: 2 involved S-boxes are in partitions 1 and 2;

Case 2.2: 2 involved S-boxes are in partitions 1 and 3;

Case 2.3: 2 involved S-boxes are in partitions 1 and 4;

Case 2.4: 2 involved S-boxes are in partitions 2 and 3;

Case 2.5: 2 involved S-boxes are in partitions 2 and 4;

Case 2.6: 2 involved S-boxes are in partitions 3 and 4;

Case 1: Without loss of generality, assume that 2 S-boxes are in partition 1 and these 2 S-boxes are $S1'$ and $S2'$.

By the definition of the linear transformation, we immediately have

$$S'_{1,1} = Q_1^{12} \oplus S_{1,1},$$

$$S'_{1,2} = Q_2^{34} \oplus S_{9,2},$$

$$S'_{1,3} = Q_3^{41} \oplus S_{13,3},$$

$$S'_{1,4} = Q_4^{23} \oplus S_{5,4},$$

$$S'_{2,1} = Q_1^{12} \oplus S_{2,1},$$

$$S'_{2,2} = Q_2^{34} \oplus S_{10,2},$$

$$S'_{2,3} = Q_3^{41} \oplus S_{14,3},$$

and

$$S'_{2,4} = Q_4^{23} \oplus S_{6,4}.$$

According to Lemma 4.4, if $S'_{1,i}$ and $S'_{2,i}$ are not used in the derivation of a linear transformation simultaneously, the number of involved S-boxes in the previous round must be at least 7. When $S'_{1,i}$ and $S'_{2,i}$ appear in the derivation of a linear approximation at the same time, the number of involved S-boxes in the previous round must be even. This is proved as follows:

Let $S'_{1,i}$ and $S'_{2,i}$ be used in the derivation of a linear approximation. It is not hard to see that, the XORed value of these 2 bits will generate 2 bits that belong to 2 different S-boxes, and when i is varied, the resulting 2 bits will change to belong to 2 other S-boxes. For example, $S'_{1,1} \oplus S'_{2,1} = S_{1,1} \oplus S_{2,1}$ and $S'_{1,2} \oplus S'_{2,2} = S_{9,2} \oplus S_{10,2}$. Obviously, $S_{1,1}$, $S_{2,1}$, $S_{9,2}$, and $S_{10,2}$ belong to 4 different S-boxes in the previous round.

This implies that when $S'_{1,i}$ and $S'_{2,i}$ are used in a linear approximation at the same time, the possible number of involved S-boxes in the previous round is 2×1, 2×2, 2×3, or 2×4, depending on whether 1, 2, 3, or 4 values of i are involved.

Summarizing the above arguments, we have proven that the lemma is true for case 1.

Case 2.1: Suppose S-boxes Si' and Sj' are from partitions 1 and 2, respectively.

In accordance with the definition of the linear transformation, it follows that:

$$S'_{i,1} = Q_1^{12} \oplus S_{i,1},$$

$$S'_{i,2} = Q_2^{34} \oplus S_{(i+8),2},$$

$$S'_{i,3} = Q_3^{41} \oplus S_{(i+12),3},$$

$$S'_{i,4} = Q_4^{23} \oplus S_{(i+4),4},$$

$$S'_{j,1} = Q_1^{12} \oplus S_{j,1},$$

$$S'_{j,2} = Q_2^{34} \oplus S_{(j+8),2},$$

$$S'_{j,3} = Q_3^{23} \oplus S_{(j+4),3},$$

and

$$S'_{j,4} = Q_4^{41} \oplus S_{(j-4),4}.$$

Since $i \in \{1,2,3,4\}$, $i+8 \in \{9,10,11,12\}$, $j \in \{5,6,7,8\}$, and $j+8 \in \{13,14,15,16\}$, it is easy to see that the 2 bits involved in $S'_{i,1} \oplus S'_{j,1}$ i.e., $S_{i,1}$ and $S_{j,1}$ will be different from that yielded from $S'_{i,2} \oplus S'_{j,2}$ i.e., $S_{(i+8),2}$ and $S_{(j+8),2}$. So, when $S'_{i,1} \oplus S'_{j,1}$ or $S'_{i,2} \oplus S'_{j,2}$ is used in the derivation of a linear approximation, there are 2 S-boxes in the previous round to be involved in the linear approximation, and when $S'_{i,1} \oplus S'_{j,1}$ and $S'_{i,2} \oplus S'_{j,2}$ are used in the derivation of the linear approximation, there are 4 S-boxes in the previous round to be involved in the linear approximation.

If at least one of $S'_{i,3}, S'_{i,4}, S'_{j,3}$ and $S'_{j,4}$ is used in the derivation of a linear approximation, since neither $S'_{i,3}$ and $S'_{j,3}$ have common component bits nor do $S'_{i,4}$ and $S'_{j,4}$, any one of 7 bits that constitutes $S'_{i,3}$, $S'_{i,4}$, $S'_{j,3}$ or $S'_{j,4}$ can not be cancelled during the derivation. Hence at least 7 S-boxes in the previous round will be involved in the linear approximation.

Combining Lemma 4.4 and the above arguments, we have shown that, when 2 S-boxes from partition 1 and 2 are involved in a linear approximation, the number of involved S-boxes in the previous round must be 2, 4, or greater than or equal to 7.

Case 2.2: For this case, assume Si' belongs to partition 1 and Sk' belongs to partition 3.

In the light of the definition of the linear transformation, we have

$$S'_{i,1} = Q_1^{12} \oplus S_{i,1},$$

$$S'_{i,2} = Q_2^{34} \oplus S_{(i+8),2},$$

$$S'_{i,3} = Q_3^{41} \oplus S_{(i+12),3},$$

$$S'_{i,4} = Q_4^{23} \oplus S_{(i+4),4},$$

$$S'_{k,1} = Q_1^{34} \oplus S_{k,1},$$

$$S'_{k,2} = Q_2^{12} \oplus S_{(k-8),2},$$

$$S'_{k,3} = Q_3^{23} \oplus S_{(k-4),3},$$

and

$$S'_{k,4} = Q_4^{41} \oplus S_{(k+4),4}. \tag{4.8}$$

Since, Q_1^{12} has no common term with Q_1^{34} and neither does Q_2^{34} with Q_2^{12}, Q_3^{41} with Q_3^{23}, and Q_4^{23} with Q_4^{41}, the combination of S-boxes Si' and Sk' must cause at least 14 S-boxes in the previous round to be involved in a linear approximation. Therefore the lemma is true in this case.

It is simple to verify cases 2.3, 2.4 and 2.6 using the same argument as for case 2.1, and case 2.5 using the argument of case 2.2. Hence, the lemma is proven. □

Lemma 4.6 *If 2 S-boxes in round r are involved in the derivation of a linear approximation such that 4 S-boxes in the previous round are involved, then each of the 4 S-boxes in round $r-1$ must only offer one output bit for the derivation of the linear approximation.*

Proof. In the proof of Lemma 4.5, we see that if the number of involved S-boxes in the round $r-1$ is 4, then the bits offered by 2 S-boxes in round r must appear in pairs in terms of their

bit positions, i.e., one involved bit of one S-box must have the same bit position of one involved bit of another S-box. Moreover, suppose Si and Sj in round r are involved in the derivation of a linear approximation. According to the definition of the linear transformation, it is not hard to see that, $S'_{i,u} \oplus S'_{j,u}$ either produces 14 bits or produces 2 bits, where $u \in \{1, 2, 3, 4\}$.

If 14 bits are produced, then these 14 bits can not be cancelled by other bits since the bit positions of these 14 bits are different from those of other bits generated by $S'_{i,v} \oplus S'_{j,v}$, where $v \neq u$. This means there will be at least 14 S-boxes in round $r-1$ to be used in the derivation of a linear approximation. It is assumed that 4 S-boxes in round $r-1$ are involved in a linear approximation, thus any $S'_{i,u} \oplus S'_{j,u}$ must produce only 2 bits.

Also, by the definition of the linear transformation, when $u \neq v$, the 2 S-boxes that connect to the 2 bits generated from $S'_{i,u} \oplus S'_{j,u}$ will be different from the 2 S-boxes that connect to the 2 bits yielded from $S'_{i,v} \oplus S'_{j,v}$.

So, only when each of 2 S-boxes in round r offers 2 input bits for the derivation of a linear approximation, the number of involved S-boxes in the round $r-1$ can be 4.

Correspondingly, the number of the total bits generated by 2 terms that are in the form of $S'_{i,u} \oplus S'_{j,u}$ is 4. These 4 bits are connected to the 4 S-boxes in round $r-1$, each of which connects to one of the 4 bits. Therefore we can equivalently say that the 4 S-boxes in round $r-1$ only offers one output bit for the derivation of the linear approximation. □

As we shall see in the proof of Theorem 4.1, combining Lemmas 4.5 and 4.7 together confirms statement (ii). Now we give Lemma 4.7 before our statement and proof of Theorem 4.1.

Lemma 4.7 *If 2 S-boxes in round r are involved in a linear approximation such that only 2 S-boxes in the previous round are involved, then each of these 2 S-boxes in round r must only offer an input bit for the derivation of the linear approximation.*

Proof. By Lemma 4.4, if the number of involved S-boxes in the previous round is 2, then the bits offered by 2 S-boxes in round r must appear in pairs with respect to their bit positions.

Moreover, suppose Si and Sj in round r are involved in the derivation of a linear approximation. According to the definition of the linear transformation, it is easy to see that, if $S'_{i,u} \oplus S'_{j,u}$ or

$S'_{i,v} \oplus S'_{j,v}$ can produce 2 bits, when $u \neq v$, the 2 S-boxes that connect to the 2 bits generated from $S'_{i,u} \oplus S'_{j,u}$ will be different from the 2 S-boxes that connect to the 2 bits yielded from $S'_{i,v} \oplus S'_{j,v}$, where $u, v \in \{1, 2, 3, 4\}$. Thus, only when each of 2 S-boxes in round r offers 1 input bit for the derivation of a linear approximation, the number of involved S-boxes in the previous round can be 2. □

Until now we have proved the lemmas with regard to the number of involved S-boxes in the previous round. By noticing Lemma 4.2 which states that the inverse linear transformation is exactly the same as the original one, all the arguments for the proof of the number of involved S-boxes in the previous round can be applied to the number of involved S-boxes in the next round. So the number of involved S-boxes in the next round satisfies the same rules as that in the previous round.

Theorem 4.1 *When an SPN adopts the designed linear transformation of Table 4.1, the average per round number of equivalent 2-term S-boxes involved in a linear approximation is at least 1.5 (except for the first and last round).*

Proof: We shall prove the theorem by combining 3 cases.

Case 1: Suppose one S-box in one round is involved in a linear approximation.

According to Lemma 4.3 and the equivalence of the linear transformation and its inverse, the number of involved S-boxes in either the previous or next round must be at least 7. Hence we only need to consider 2 consecutive rounds. Since there are at least $(1 + 7)$ involved S-boxes in 2 rounds, and for any S-box its number of equivalent 2-term S-boxes is at least 0.5, the per round number of equivalent 2-term S-boxes is $\geq \frac{(1+7)\times 0.5}{2} = 2.0$. Hence, the theorem is true for this case.

Case 2: Suppose 2 S-boxes in one round are involved in a linear approximation.

By Lemma 4.6, in this scenario the number of involved S-boxes in the previous or next round must be 2, 4, 6, or not less than 7.

(i) Suppose the number of involved S-boxes in either the previous round or the next round is 2, as in Lemma 4.7. Each of the 2 S-boxes in this round must offer only one input bit and one

output bit to the derivation of the linear approximation. Thus, for each of these 2 S-boxes its equivalent number of 2-term S-boxes is 1. Consequently, for this round the per round number of equivalent 2-term S-boxes is 2.

(ii) Suppose the number of S-boxes in round r is 2, the number of S-boxes in round $r-1$ is 4, and the number of S-boxes in round $r-2$ is 2. Then by Lemma 4.6, each of the 4 S-boxes in round $r-1$ must offer only one input bit and one output bit to the derivation of the linear approximation. This means for these 4 S-boxes, the equivalent number of 2-term S-boxes for each is 1. Meanwhile, for every S-box in round r or $r-2$, its equivalent number of 2-term S-boxes is 0.5. Hence for this 3 rounds, the per round number of equivalent 2-term S-boxes is $\frac{4\times1+0.5\times4}{3} = 2$.

(iii) Suppose the number of S-boxes in round r is 2, the number of S-boxes in round $r-1$ is 4, and the number of S-boxes in round $r-2$ is 3 or more. In this scenario, we only need to calculate the per round number of equivalent 2-term S-boxes for rounds r and $r-1$. For each of the $(2+4)$ S-boxes in round r and $r-1$, its number of equivalent 2-term S-boxes is at least 0.5. So the per round number of equivalent 2-term S-boxes is at least $\frac{(2+4)\times0.5}{2} = 1.5$.

By Lemma 4.2, the inverse linear transformation is exactly same as original. All the arguments for (ii) and (iii) are applicable when S-boxes appear in reverse order. So, combining all of these cases, it is not hard to see that, no matter what round (except the first and the last round) 2 S-boxes appear in, the per round number of equivalent 2-term S-boxes for a section of linear path related to these 2 S-boxes is never less than 1.5.

Case 3: Suppose 3 or more S-boxes in one round are involved in a linear approximation. Since for any S-box its equivalent number of 2-term S-boxes is not less than 0.5, the per round number of equivalent 2-term S-boxes for this round is $\geq 3 \times 0.5 = 1.5$.

When the per round number of equivalent 2-term S-boxes involved in a linear approximation is calculated, the above 3 cases should be considered together. By joining the 3 cases together, the theorem follows. □

4.6 Effectiveness of the Linear Transformation

In this section the effectiveness of the designed linear transformation is considered.

Suppose a cryptanalyst uses basic linear cryptanalysis (algorithm 1 in [16]) to attack an 12-round SPN that employs the designed linear transformation. According to [16] the approximate number of known plaintexts required to guess a correct equivalent key bit with a success rate of 97.7% is $N_L = |p_L - 1/2|^{-2}$, with $|p_L - 1/2| = 1/2 \prod_{i=1}^{\delta} |2p_i - 1|$, where δ is the total number of S-boxes involved in a linear approximation.

According to the definition $2\eta_i(\gamma) = |2p_i(\gamma) - 1|$, where γ is the number of bits included in a linear approximation of an S-box, and the definition for the equivalent number of 2-term S-boxes, $|p_L - 1/2|$ can be bounded by $|p_L - 1/2| \leq 1/2(2\eta(2))^{\sigma}$, where σ is the equivalent number of 2-term S-boxes involved in a linear approximation and $\eta(2)$ is upper bound on $\eta_i(2)$.

On the basis of Theorem 4.1, for an 12 round SPN, the number of equivalent 2-term S-boxes involved in a linear approximation is at least $(10 \times 1.5 + 2 \times 1)$, where 10 refers to the number of rounds excluding the first and last rounds, 1.5 is the bound for the average per round number of equivalent 2-term S-boxes in the middle rounds, and 1 is the bound for the first and last rounds. Thus $N_L = |p_L - 1/2|^{-2} \geq [1/2(2\eta(2))^{\sigma}]^{-2} \geq [1/2(1/4)^{(10\times1.5+2\times1)}]^2 = 2^{70}$. A 12 round SPN that adopts the linear transformation has 2^{70} known plaintext-ciphertext pairs required for the basic linear attack. Given that only 2^{64} plaintexts are available, it is clear that such a cipher is secure. For comparison, a cipher which uses a permutation or the linear transformation of (4.5) with S-boxes that satisfy the nonlinearity requirement of (4.2), as few as 2^{50} known plaintext/ciphertext pairs are required, by the conclusions about the per round number of equivalent 2-term S-boxes in Section 4.4. Therefore the linear transformation of Table 4.1 is very effective in strengthening an SPN against linear cryptanalysis.

4.7 Conclusion

In this chapter we proposed a novel linear transformation for the interconnection of S-boxes. The linear transformation is designed to be used in an SPN that consists of 16 4×4 S-boxes in each round. By utilizing the linear transformation, the per round number of equivalent 2-term

S-boxes involved in an overall linear approximation is raised to at least 1.5. As well, combining this lower bound on the per round number of equivalent 2-term S-boxes involved in a cipher linear approximation and new S-box nonlinear properties as in (4.2) gives a new higher lower bound on the complexity of linear cryptanalysis.

Chapter 5

Security Against Differential Cryptanalysis

In Chapter 4, to resist linear cryptanalysis, we proposed a new linear transformation, and proved that by using the linear transformation and a new nonlinearity constraint the per round number of equivalent 2-term S-boxes involved in a linear approximation is at least 1.5. Another powerful attack on block ciphers is differential cryptanalysis. In this chapter, the security against differential cryptanalysis for an SPN that adopts the linear transformation of Table 4.1 is discussed. It is established that, when the linear transformation is used and S-boxes are selected which satisfy a diffusion order of 1, the average number of S-boxes involved per round in a differential characteristic is at least 3.

5.1 Average Number of S-boxes Involved in a One-round Characteristic

One powerful attack on DES-like ciphers is the method of differential cryptanalysis introduced by Biham and Shamir [3]. This is a chosen-plaintext attack. In the attack on an R-round block cipher, an $(R-1)$-round differential characteristic can be used, which describes the correlation between the input and output differences for consecutive $(R-1)$ rounds. Suppose two messages, M' and M'', are input to one round of an SPN at different times, we use the notation of ΔM to represent the bit-wise XOR difference of the two messages, i.e., $\Delta M = M' \oplus M''$. An r-round characteristic is defined as a series of differential pairs: $\Omega_r = \{(\Delta U_1, \Delta V_1), \ldots, (\Delta U_r, \Delta V_r)\}$, where ΔU_i and ΔV_i refer to the input and output difference of a particular round i, respectively.

The probability of a *one-round differential characteristic* is defined to be the conditional probability that, given some particular difference in the inputs to the round, some particular difference in the outputs of that round is achieved. Assume that the inputs are independent between rounds. This assumption is satisfied if the round keys are mutually independent. Then, since an r-round characteristic can be viewed as the concatenation of r one-round characteristics, the probability of an r-round characteristic is obtained by multiplying the r probabilities of one-round characteristics.

Let $p(\Delta U_i, \Delta V_i)$ represent the probability of the i-th round pair, and let p_{Ω_r} represent the probability of an r-round characteristic. Then $p_{\Omega_r} = \prod_{i=1}^{r} p(\Delta U_i, \Delta V_i)$. For an R-round SPN, the number of chosen plaintext-ciphertext pairs required for differential cryptanalysis can be approximated by $N_D = \frac{1}{p_{\Omega_{R-1}}}$ [3], where $p_{\Omega_{R-1}}$ is the probability of the best $(R-1)$-round characteristic.

Assume the average number of S-boxes involved in a one-round characteristic that constitutes an $(R-1)$-round characteristic is n_a, and the maximum S-box XOR pair probability is p_δ, where $p_\delta = \frac{M_\oplus}{2^n}$, with $M_\oplus$ representing the maximum entry in the XOR distribution tables of the $n \times n$ S-boxes used in the SPN under consideration. Then the number of chosen plaintext/ciphertext pairs required in the attack, N_D, is given by

$$N_D \geq \frac{1}{(p_\delta)^{(R-1)\cdot n_a}}. \tag{5.1}$$

In the context of an R-round SPN, p_δ may be viewed as a constant because all of the S-boxes used in SPNs can be selected such that they satisfy the same criteria. Hence n_a is the sole factor that affects the value of N_D.

Later in this chapter, we focus on the discussion of the bound for n_a, the average number of S-boxes involved in a one-round characteristic.

5.2 Selection of S-boxes

As in the previous chapter, we are still considering the 64-bit substitution-permutation network consisting of 4×4 S-boxes. As proposed in Chapter 4, the combining S-boxes are selected such that they satisfy two criteria: 1) For the differential property, the diffusion order of an S-box is

$\lambda = 1$, where diffusion order is described in Chapter 3; 2) For the nonlinearity property, $2\eta(\gamma)$, defined in Chapter 3, satisfies (4.2).

5.3 Strength of Previously Proposed Linear Transformations

In this section, we shall analyse the average number of S-boxes involved in a one-round characteristic of an SPN for which one of former linear transformations is applied as the interconnection between rounds of S-boxes.

Lemma 5.1 *If an SPN is constructed such that: 1) it uses the kind of permutation where each of the outputs of an S-box go to different S-boxes in the next round, 2) the number of S-boxes in each round equals the size of an S-box and 3) the diffusion order of all S-boxes is $\lambda = 1$, then the average number of S-boxes involved in a one-round characteristic can be 1.5.*

Proof: The proof of the lemma is done by considering the case illustrated in Figure 5.1. In Figure 5.1, all of the S-boxes satisfy $\lambda = 1$, the bold lines represent paths of bit changes of S-boxes. It can be seen that in every 4 rounds there are 6 S-boxes involved in an iterative differential characteristic, thus the average number of S-boxes involved in a one-round characteristic is 1.5.

□

Lemma 5.2 *If an SPN uses Ayoub's permutation and uses S-boxes for which $\lambda = 1$, then the average number of S-boxes involved in a one-round characteristic can be 2.0.*

Proof: We prove by giving an example. In Figure 5.2, all of the S-boxes satisfy $\lambda = 1$, and the bold lines represent paths of bit changes of S-boxes. The bold lines thus constitute the path of a differential characteristic. For this characteristic, in each round, there are 2 S-boxes involved. Thus in this scenario the average number of S-boxes involved in a one-round characteristic is 2.0. □

The linear transformation suggested in [12] is defined by $V = \pi[L(u)]$, where $V = [V_1, V_2, \ldots, V_N]$ is the vector of input bits of a round of S-boxes, $U = [U_1, U_2, \ldots, U_N]$ is the vector of bits from

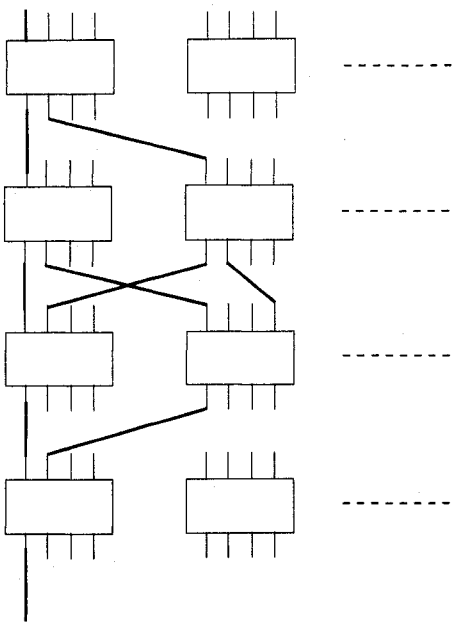


Figure 5.1: A characteristic in an SPN using permutation

the previous round output, π is a permutation for which no two outputs of an S-box are connected to one S-box in the next round, and $L(u) = [L_1(U), \ldots, L_N(U)]$, where $L_i(U) = U_1 \oplus U_2 \oplus U_{i-1} \oplus U_{i+1} \ldots U_N$.

Lemma 5.3 *When an SPN is combined with the linear transformation in [12] and uses* 4×4 *S-boxes that have diffusion order* $\lambda = 1$*, the lower bound for the average number of S-boxes involved in a one-round characteristic can not be expected to exceed 2.*

Proof. If we can give a scenario where the average number of S-boxes involved in a one-round characteristic is 2, the lemma is proven.

In the definition of the linear transformation, π is any permutation for which no two output bits of an S-box are connected to one S-box in the next round. Hence, Ayoub's permutation [2] satisfies this condition and can be used in the linear transformation, where Ayoub's permutation is illustrated in Figure 5.2.

Suppose we use the bold path representing bit changes to achieve a differential characteristic. In accordance with the definition of the linear transformation, we immediately have these relations: $S'_{1,1} = S_{1,1} \oplus Q$, $S'_{1,2} = S_{2,1} \oplus Q$, $S'_{5,1} = S_{1,2} \oplus Q$, and $S'_{5,2} = S_{2,2} \oplus Q$, where $Q = U_1 \oplus U_2 \oplus \ldots \oplus U_{N-1} \oplus U_N = S_{1,1} \oplus S_{1,2} \oplus S_{1,3} \oplus S_{1,4} \oplus \ldots S_{16,1} \oplus S_{16,2} \oplus S_{16,3} \oplus S_{16,4}$, where $S_{i,j}$ represents

an output bit of an S-box in one round, and $S'_{i,j}$ represents an input bit of an S-box in the next round.

Assume that $\Delta S_{1,1} = 1$, $\Delta S_{1,2} = 1$, $\Delta S_{2,1} = 1$ and $\Delta S_{2,2} = 1$. Then $\Delta Q = \Delta U_1 \oplus \Delta U_2 \oplus \ldots \oplus \Delta U_N = \Delta S_{1,1} \oplus \Delta S_{1,2} \oplus \ldots \oplus \Delta S_{2,1} \oplus \Delta S_{2,2} \oplus \ldots \oplus \Delta S_{16,3} \oplus S_{16,4} = 1 \oplus 1 \oplus \ldots \oplus 1 \oplus 1 \oplus \ldots \oplus 0 \oplus 0 = 0$.

Consequently, $\Delta S'_{1,1} = \Delta S_{1,1} \oplus \Delta Q = 1$, $\Delta S'_{1,2} = \Delta S_{2,1} \oplus \Delta Q = 1$, $\Delta S'_{5,1} = \Delta S_{1,2} \oplus \Delta Q = 1$, and $\Delta S'_{5,2} = \Delta S_{2,2} \oplus \Delta Q = 1$. Similarly, according to the connections of the highlighted lines in Figure 5.2, when $\Delta S_{1,1} = 1$, and $\Delta S_{5,1} = 1$, we have $\Delta S'_{1,1} = \Delta S_{1,1}$ and $\Delta S'_{2,1} = \Delta S_{5,1}$.

By these difference expressions, it is clear that a differential characteristic that involves S-boxes $S1, S2, S1', S5'$ in every two rounds exists. Hence, in this case the average number of S-boxes involved in a one-round characteristic is $4.0/2 = 2.0$. Notice that the average number of S-boxes involved in a one-round chatacteistic could be fractional. □

The linear transformation presented in [32] can be described by

$$S'_{i,j} = \bigoplus_{k=1}^{m} S_{k,j} \oplus S_{i,j} \tag{5.2}$$

where m represents the number of S-boxes in one round of an SPN, $S'_{i,j}$ denotes the j-th input bit of the i-th S-box in one round, and $S_{k,j}$ refers to the j-th output bit of the k-th S-box in the previous round. For an SPN that consists of S-boxes satisfying diffusion order of 1, this linear transformation can not be expected to have the lower bound for the average number of S-boxes involved in a one-round characteristic to be greater than 2.0.

Lemma 5.4 *By using the linear transformation represented by (5.2) and S-boxes satisfying $\lambda = 1$, the average number of S-boxes involved in a one-round characteristic can be 2.0.*

Proof: Without loss of generality, suppose a cryptanalyst adopts S-boxes $S1$ and $S2$ to obtain a differential characteristic. First we consider the difference path in the first round. From the definition of the linear transformation, we have

$$S'_{1,1} = \oplus_{k=1}^{m} S_{k,1} \oplus S_{1,1},$$

$$S'_{2,1} = \oplus_{k=1}^{m} S_{k,1} \oplus S_{2,1},$$

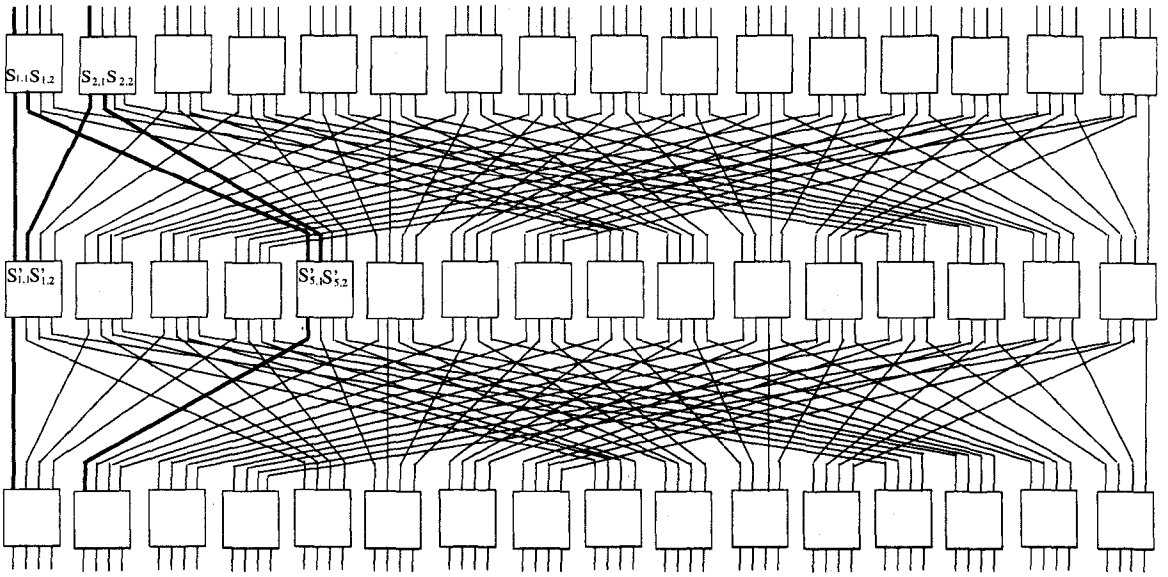


Figure 5.2: A differential characteristic in an SPN using permutation

$$S'_{1,2} = \oplus_{k=1}^{m} S_{k,2} \oplus S_{1,2},$$

and

$$S'_{2,2} = \oplus_{k=1}^{m} S_{k,2} \oplus S_{2,2}.$$

It follows that

$$\Delta S'_{1,1} = \oplus_{k=1}^{m} \Delta S_{k,1} \oplus \Delta S_{1,1},$$

$$\Delta S'_{2,1} = \oplus_{k=1}^{m} \Delta S_{k,1} \oplus \Delta S_{2,1},$$

$$\Delta S'_{1,2} = \oplus_{k=1}^{m} \Delta S_{k,2} \oplus \Delta S_{1,2},$$

and

$$\Delta S'_{2,2} = \oplus_{k=1}^{m} \Delta S_{k,2} \oplus \Delta S_{2,2}.$$

Letting $\Delta S_{1,1} = \Delta S_{1,2} = \Delta S_{2,1} = \Delta S_{2,2} = 1$, and $\Delta S_{k,1} = \Delta S_{k,2} = 0$ for $k \geq 3$, then

$$\oplus_{k=1}^{m} \Delta S_{k,1} = 1 \oplus 1 \oplus 0 \oplus \ldots \oplus 0 = 0,$$

and

$$\oplus_{k=1}^{m} \Delta S_{k,2} = 1 \oplus 1 \oplus 0 \oplus \ldots \oplus 0 = 0.$$

Hence

$$\Delta S'_{1,1} = \oplus_{k=1}^{m} \Delta S_{k,1} \oplus \Delta S_{1,1} = 0 \oplus 1 = 1,$$

$$\Delta S'_{2,1} = \oplus_{k=1}^{m} \Delta S_{k,1} \oplus \Delta S_{2,1} = 0 \oplus 1 = 1,$$

$$\Delta S'_{1,2} = \oplus_{k=1}^{m} \Delta S_{k,2} \oplus \Delta S_{1,2} = 0 \oplus 1 = 1,$$

and

$$\Delta S'_{2,2} = \oplus_{k=1}^{m} \Delta S_{k,2} \oplus \Delta S_{2,2} = 0 \oplus 1 = 1.$$

Now consider the difference path in the next round. Similarly, letting $\Delta S_{1,1} = \Delta S_{2,1} = 1$, then

$$\oplus_{k=1}^{m} \Delta S_{k,1} = 1 \oplus 1 \oplus 0 \oplus \ldots \oplus 0 = 0.$$

Hence,

$$\Delta S'_{1,1} = \oplus_{k=1}^{m} \Delta S_{k,1} \oplus \Delta S_{1,1} = 0 \oplus 1 = 1,$$

and

$$\Delta S'_{2,1} = \oplus_{k=1}^{m} \Delta S_{k,1} \oplus \Delta S_{2,1} = 0 \oplus 1 = 1.$$

Thus in every round S-boxes $S1$ and $S2$ can be involved in the differential characteristic, and in this case the average number of S-boxes involved in an one-round characteristic is $4.0/2 = 2.0$.

□

We have now completed the analysis of the strength of previous proposed linear transformation against differential cryptanalysis and will now turn our attention to the analysis of the linear transformation proposed in Chapter 4.

5.4 Lower Bound on the Number of S-boxes

We shall prove a theorem which demonstrates that by using our linear transformation the lower bound for the average number of S-boxes involved in a one-round characteristic is 3. The detailed description of the linear transformation is given Chapter 4. The theorem is going to be proven by showing that the following statements are true and then combining them.

(i) If one S-box in one round is involved in a differential characteristic, then the number of involved S-boxes in the next round must be at least 7;

(ii) If 2 S-boxes in one round are involved in a differential characteristic, and at least one of them has 2 or more outputs involved, then in the next round the possible number of involved S-boxes is 4, 6, or greater than or equal to 7;

(iii) If 4 S-boxes in one round are involved in a differential characteristic, and at least one of them has 2 or more inputs involved in the characteristic, then the number of involved S-boxes in the previous round can not be 2.

Lemma 5.5 *If one S-box in one round is involved in a differential characteristic, then the number of involved S-boxes in the next round must be* 7.

Proof. Without loss of generality, suppose S-box $S1$ is involved in a differential characteristic. Then, by the definition of the linear transformation, the 7 bits in the next round each of which contains $S_{1,1}$ in its component bits are:

$$S'_{i,1} = Q_1^{12} \oplus S_{i,1}$$

where $2 \leq i \leq 8$, and the notation of Q_k^{ij} is defined in the definition of the linear transformation in Chapter 4.

The 7 bits in the next round, each of which contains $S_{1,2}$ in its components, are:

$$S'_{i,2} = Q_2^{12} \oplus S_{(i-8),2}$$

where $10 \leq i \leq 16$.

The 7 bits in the next round, each of which contains $S_{1,3}$ in its component bits, are:

$$S'_{i,3} = Q_3^{41} \oplus S_{(i+12),3}$$

where $1 \leq i \leq 4$ and

$$S'_{i,3} = Q_3^{41} \oplus S_{(i-12),3},$$

where $14 \leq i \leq 16$.

And the 7 bits in the next round, each of which contains $S_{1,4}$ in its component bits, are:

$$S'_{i,4} = Q_4^{41} \oplus S_{(i-4),4},$$

where $6 \leq i \leq 8$ and

$$S'_{i,4} = Q_4^{41} \oplus S_{(i+4),4},$$

where $9 \leq i \leq 12$.

From these expressions, we obtain (1) when $\Delta S_{1,j} = 1$, the 7 bits that include $S_{1,j}$ will be changed, and (2) the 7 bits that include $S_{1,j}$ and the 7 bits that include $S_{1,k}$ where $k \neq j$, do not have common terms. Hence, when $\Delta S_{1,j} = 1$ and $\Delta S_{1,k} = 1$, where $k \neq j$, these 2 sets of 7 bits will change independently without interfering with each other.

Since it is assumed that $S1$ is involved in a characteristic, this means that $\Delta S_{1,j} = 1$, for at least one value of j, $j \in \{1, 2, 3, 4\}$. Thus at least 7 bits that include $S_{1,j}$ will change. By the definition of the transformation, these 7 bits belong to 7 different S-boxes. Hence at least 7 S-boxes in the next round will be involved in the characteristic. □

Lemma 5.6 *If, in one round for a specific k, only one $\Delta S_{i,k}$ is involved in a characteristic (i.e. $\Delta S_{i,k} = 1$), but none of other $\Delta S_{j,k}$ is involved (i.e. $\Delta S_{j,k} = 0$), where $1 \leq i, j \leq 16$, $i \neq j$, $1 \leq k \leq 4$, then the number of involved S-boxes in the next round must be at least 7 .*

Proof: Without loss of generality, suppose $\Delta S_{1,1} = 1$, but $\Delta S_{j,1} = 0$, $2 \leq j \leq 16$.

By the definition of the linear transformation, there are 7 bits that include $S_{1,1}$ as their component bits, they are

$$S'_{j,1} = Q_1^{12} \oplus S_{j,1},$$

where $2 \leq j \leq 8$.

Also, based on the expression $Q_1^{12} = S_{1,1} \oplus S_{2,1} \oplus S_{3,1} \oplus S_{4,1} \oplus S_{5,1} \oplus S_{6,1} \oplus S_{7,1} \oplus S_{8,1}$ defined in the linear transformatiom and the assumption that $\Delta S_{1,1} = 1$ and $\Delta S_{j,1} = 0, 2 \leq j \leq 16$, these expressions immediately follow:

$$\begin{aligned} \Delta Q_1^{12} &= \Delta S_{1,1} \oplus \Delta S_{2,1} \oplus \Delta S_{3,1} \oplus \Delta S_{4,1} \oplus \Delta S_{5,1} \oplus \Delta S_{6,1} \oplus \Delta S_{7,1} \oplus \Delta S_{8,1} \\ &= 1 \oplus 0 \oplus 0 \oplus 0 \oplus 0 \oplus 0 \oplus 0 \oplus 0 \\ &= 1, \end{aligned}$$

and

$$\begin{aligned} \Delta S'_{j,1} &= \Delta Q_1^{12} \oplus \Delta S_{j,1}, \\ &= 1 \end{aligned}$$

where $2 \leq j \leq 8$. Thus all of the 7 bits $S'_{j,1}, 2 \leq j \leq 8$ are changed, i.e. $\Delta S'_{j,1} = 1$, and these 7 bits belong to 7 different S-boxes in the next round, according to the definition of the linear transformation.

In addition, if the bits $S_{i,l}$, $1 \leq i \leq 16$, $1 \leq l \leq 4$, other than $S_{i,k}$, $1 \leq i \leq 16$, $1 \leq k \leq 4$, where $l \neq k$, in this round are involved in a characteristic, since bit k of an S-box can not cancel bit l, the number of involved S-boxes in the next round will not decrease. So in the next round the number of S-boxes is at least 7. □

Lemma 5.7 *If 2 S-boxes in one round are involved in a differential characteristic, and at least one of them has 2 or more outputs involved, then in the next round the possible number of involved S-boxes is 4, 6, or greater than or equal to 7.*

Proof. The proof of the lemma is done by considering 2 cases: (1) the 2 involved S-boxes in one round are in the same partition; and (2) the 2 involved S-boxes belong to two different partitions.

For case 2, there are 6 subcases to be considered:

Case 2.1: 2 involved S-boxes are in partition 1 and 2;

Case 2.2: 2 involved S-boxes are in partition 1 and 3;

Case 2.3: 2 involved S-boxes are in partition 1 and 4;

Case 2.4: 2 involved S-boxes are in partition 2 and 3;

Case 2.5: 2 involved S-boxes are in partition 2 and 4;

Case 2.6: 2 involved S-boxes are in partition 3 and 4.

Case 1: Without loss of generality, let us assume that the 2 involved S-boxes are in partition 1 and these 2 S-boxes are $S1$ and $S2$.

According to Lemma 5.6, if $\Delta S_{1,i} = 1$ and $\Delta S_{2,i} = 1$, where $1 \leq i \leq 4$, do not appear in a characteristic simultaneously, the number of involved S-boxes in the next round must be at least 7.

Besides, by the definition of the linear transformation, we have

$$S'_{i,1} = Q_1^{12} \oplus S_{i,1},$$

where $1 \leq i \leq 8$,

$$S'_{i,2} = Q_2^{12} \oplus S_{(i-8),2},$$

where $9 \leq i \leq 16$,

$$S'_{i,3} = Q_3^{41} \oplus S_{(i+12),3},$$

where $1 \leq i \leq 4$,

$$S'_{i,3} = Q_3^{41} \oplus S_{(i-12),3},$$

where $13 \leq i \leq 16$,

$$S'_{i,4} = Q_4^{41} \oplus S_{(i-4),4},$$

where $5 \leq i \leq 8$, and

$$S_{i,4'} = Q_4^{41} \oplus S_{(i+4),4},$$

where $9 \leq i \leq 12$.

Since "$\oplus$" operation is linear, all of these expressions can be changed directly into difference form. Hence,

$$\Delta S'_{i,1} = \Delta Q_1^{12} \oplus \Delta S_{i,1},$$

where $1 \leq i \leq 8$,

$$\Delta S'_{i,2} = \Delta Q_2^{12} \oplus \Delta S_{(i-8),2},$$

where $9 \leq i \leq 16$,

$$\Delta S'_{i,3} = \Delta Q_3^{41} \oplus \Delta S_{(i+12),3},$$

where $1 \leq i \leq 4$,

$$\Delta S'_{i,3} = \Delta Q_3^{41} \oplus \Delta S_{(i-12),3},$$

where $13 \leq i \leq 16$,

$$\Delta S'_{i,4} = \Delta Q_4^{41} \oplus \Delta S_{(i-4),4},$$

where $5 \leq i \leq 8$, and

$$\Delta S'_{i,4} = \Delta Q_4^{41} \oplus \Delta S_{(i+4),4},$$

where $9 \leq i \leq 12$.

Based on these difference expressions, we obtain:

- when $\Delta S_{1,1} = 1$ and $\Delta S_{2,1} = 1$ are used in a characteristic, then $\Delta S'_{1,1} = 1$ and $\Delta S'_{2,1} = 1$, implying S-boxes $S1$ and $S2$ in the next round are involved in the characteristic;
- when $\Delta S_{1,2} = 1$ and $\Delta S_{2,2} = 1$ are used, then $\Delta S'_{9,2} = 1$ and $\Delta S'_{10,2} = 1$, implying S-boxes $S9$ and $S10$ in the next round are involved in the characteristic;
- when $\Delta S_{1,3} = 1$ and $\Delta S_{2,3} = 1$ are used in a characteristic, then $\Delta S'_{13,3} = 1$ and $\Delta S'_{14,3} = 1$, implying S-boxes $S13$ and $S14$ in the next round are involved in the characteristic;

- when $\Delta S_{1,4} = 1$ and $\Delta S_{2,4} = 1$ are employed in a characteristic, then $\Delta S'_{5,4} = 1$ and $\Delta S'_{6,4} = 1$, implying S-boxes $S5$ and $S6$ in the next round are involved in the characteristic.

Thus, when S-boxes $S1$ and $S2$ have 2 or more outputs of the same position involved in a characteristic, the number of S-boxes involved in the next round must be 4, 6, or 8.

Summarizing the above arguments, we have proven the lemma is true for case 1.

Case 2.1: Without loss of generality, assume S-boxes $S1$ and $S5$ in this round are involved in a characteristic. By the definition of the linear transformation and the similar deduction as in case 1, it follows that:

$$\Delta S'_{i,1} = \Delta Q_1^{12} \oplus \Delta S_{i,1},$$

where $1 \leq i \leq 8$,

$$\Delta S'_{i,2} = \Delta Q_2^{12} \oplus \Delta S_{(i+8),2},$$

where $9 \leq i \leq 16$,

$$\Delta S'_{i,3} = \Delta Q_3^{41} \oplus \Delta S_{(i+12),3},$$

where $1 \leq i \leq 4$,

$$\Delta S'_{i,3} = \Delta Q_3^{41} \oplus \Delta S_{(i-12),3},$$

where $13 \leq i \leq 16$,

$$\Delta S'_{i,4} = \Delta Q_4^{41} \oplus \Delta S_{(i-4),4},$$

where $5 \leq i \leq 8$,

$$\Delta S'_{i,4} = \Delta Q_4^{41} \oplus \Delta S_{(i+4),4},$$

where $9 \leq i \leq 12$,

$$\Delta S'_{i,3} = \Delta Q_3^{23} \oplus \Delta S_{(i+4),3},$$

where $5 \leq i \leq 8$,

$$\Delta S'_{i,3} = \Delta Q_3^{23} \oplus \Delta S_{(i-4),3},$$

where $9 \leq i \leq 12$,

$$\Delta S'_{i,4} = \Delta Q_4^{23} \oplus \Delta S_{(i+4),4},$$

where $1 \leq i \leq 4$, and

$$\Delta S'_{i,4} = \Delta Q_4^{23} \oplus \Delta S_{(i-4),4},$$

where $13 \leq i \leq 16$.

From these expressions, it is clearly demonstrated that:

- when $\Delta S_{1,1} = 1$ and $\Delta S_{5,1} = 1$ are used in a characteristic, then $\Delta S'_{1,1} = 1$ and $\Delta S'_{5,1} = 1$, implying S-boxes $S1$ and $S5$ in the next round are involved in the characteristic;
- when $\Delta S_{1,2} = 1$ and $\Delta S_{5,2} = 1$ are employed in a characteristic, then $\Delta S'_{9,2} = 1$ and $\Delta S'_{13,2} = 1$, implying S-boxes $S9$ and $S13$ in the next round are involved in the characteristic;
- when $\Delta S_{1,3} = 1$ and $\Delta S_{5,3} = 1$ are used in a characteristic, then $\Delta S'_{1,3} = 1$, $\Delta S'_{2,3} = 1$, $\Delta S'_{3,3} = 1$, $\Delta S'_{4,3} = 1$, $\Delta S'_{14,3} = 1$, $\Delta S'_{15,3} = 1$, $\Delta S'_{16,3} = 1$, $\Delta S'_{5,3} = 1$, $\Delta S'_{6,3} = 1$, $\Delta S'_{7,3} = 1$, $\Delta S'_{8,3} = 1$, $\Delta S'_{10,3} = 1$, $\Delta S'_{11,3} = 1$, and $\Delta S'_{12,3} = 1$, implying 14 S-boxes - $S1$, $S2$, $S3$, $S4$, $S14$, $S15$, $S16$, $S5$, $S6$, $S7$, $S8$, $S10$, $S11$ and $S12$ - in the next round are involved in the characteristic;
- when $\Delta S_{1,4} = 1$ and $\Delta S_{5,4} = 1$ are used in a characteristic, then $\Delta S'_{2,4} = 1$, $\Delta S'_{3,4} = 1$, $\Delta S'_{4,4} = 1$, $\Delta S'_{13,4} = 1$, $\Delta S'_{14,4} = 1$, $\Delta S'_{15,4} = 1$, $\Delta S'_{16,4} = 1$, $\Delta S'_{6,4} = 1$, $\Delta S'_{7,4} = 1$, $\Delta S'_{8,4} = 1$, $\Delta S'_{9,4} = 1$, $\Delta S'_{10,4} = 1$, $\Delta S'_{11,4} = 1$, and $\Delta S'_{12,4} = 1$, implying 14 S-boxes - $S2$, $S3$, $S4$, $S13$, $S14$, $S15$, $S16$, $S6$, $S7$, $S8$,$S9$, $S10$, $S11$, and $S12$ - in the next round are involved in the characteristic.

Thus when S-boxes $S1$ and $S5$ have 2 or more outputs of the same position involved in a characteristic, the number of involved S-boxes in the next round must be 4 , or at least 14.

By combining Lemma 5.6 and the above arguments, Lemma 5.7 is true for this case.

Case 2.2: Without loss of generality, let us assume $S1$ and $S9$ in this round are involved in a characteristic.

In light of the definition of the linear transformation and the deduction as in Case 1 or 2.1, we have

$$\Delta S'_{i,1}\Delta Q_1^{12} \oplus \Delta S_{i,1},$$

where $1 \leq i \leq 8$,

$$\Delta S'_{i,2} = \Delta Q_2^{12} \oplus \Delta S_{(i-8),2},$$

where $9 \leq i \leq 16$,

$$\Delta S'_{i,3} = \Delta Q_3^{41} \oplus \Delta S_{(i+12),3},$$

where $1 \leq i \leq 4$,

$$\Delta S'_{i,3}\Delta Q_3^{41} \oplus \Delta S_{(i-12),3},$$

where $13 \leq i \leq 16$,

$$\Delta S'_{i,4} = \Delta Q_4^{41} \oplus \Delta S_{(i-4),4},$$

where $5 \leq i \leq 8$,

$$\Delta S'_{i,4} = \Delta Q_4^{41} \oplus \Delta S_{(i+4),4},$$

where $9 \leq i \leq 12$,

$$\Delta S'_{i,1} = \Delta Q_1^{34} \oplus \Delta S_{i,1},$$

where $9 \leq i \leq 16$,

$$\Delta S'_{i,2} = \Delta Q_2^{34} \oplus \Delta S_{(i+8),2},$$

where $1 \leq i \leq 8$,

$$\Delta S'_{i,3} = \Delta Q_3^{23} \oplus \Delta S_{(i+4),3},$$

where $5 \leq i \leq 8$,

$$\Delta S'_{i,3} = \Delta Q_3^{23} \oplus \Delta S_{(i-4),3},$$

where $9 \leq i \leq 12$,

$$\Delta S'_{i,4} = \Delta Q_4^{23} \oplus \Delta S_{(i+4),4},$$

where $1 \leq i \leq 4$, and

$$\Delta S'_{i,4} = \Delta Q_4^{23} \oplus \Delta S_{(i-4),4},$$

where $13 \leq i \leq 16$.

Obseving these expressions, we immediately know:

- when $\Delta S_{1,1} = 1$ and $\Delta S_{9,1} = 1$ are used in a characteristic, then $\Delta S'_{2,1} = 1$, $\Delta S'_{3,1} = 1$, $\Delta S'_{4,1} = 1$, $\Delta S'_{5,1} = 1$, $\Delta S'_{6,1} = 1$, $\Delta S'_{7,1} = 1$, $\Delta S'_{8,1} = 1$, $\Delta S'_{10,1} = 1$, $\Delta S'_{11,1} = 1$, $\Delta S'_{12,1} = 1$, $\Delta S'_{13,1} = 1$, $\Delta S'_{14,1} = 1$, $\Delta S'_{15,1} = 1$, and $\Delta S'_{16,1} = 1$, implying 14 S-boxes - $S2$, $S3$, $S4$, $S5$, $S6$, $S7$, $S8$, $S10$, $S11$, $S12$, $S13$, $S14$, $S15$, and $S16$ - in the next round are involved in the characteristic;

- when $\Delta S_{1,2} = 1$ and $\Delta S_{9,2} = 1$ are used in a characteristic, then $\Delta S'_{2,2} = 1$, $\Delta S'_{3,2} = 1$, $\Delta S'_{4,2} = 1$, $\Delta S'_{5,2} = 1$, $\Delta S'_{6,2} = 1$, $\Delta S'_{7,2} = 1$, $\Delta S'_{8,2} = 1$, $\Delta S'_{10,2} = 1$, $\Delta S'_{11,2} = 1$, $\Delta S'_{12,2} = 1$, $\Delta S'_{13,2} = 1$, $\Delta S'_{14,2} = 1$, $\Delta S'_{15,2} = 1$, and $\Delta S'_{16,2} = 1$, implying 14 S-boxes - $S2$, $S3$, $S4$, $S5$, $S6$, $S7$, $S8$, $S10$, $S11$, $S12$, $S13$, $S14$, $S15$, and $S16$ - in the next round are involved in the characteristic;

- when $\Delta S_{1,3} = 1$ and $\Delta S_{9,3} = 1$ are used in a characteristic, then $\Delta S'_{1,3} = 1, \Delta S'_{2,3} = 1, \Delta S'_{3,3} = 1, \Delta S'_{4,3} = 1, \Delta S'_{14,3} = 1, \Delta S'_{15,3} = 1, \Delta S'_{16,3} = 1, \Delta S'_{6,3} = 1, \Delta S'_{7,3} = 1, \Delta S'_{8,3} = 1$, $\Delta S'_{9,3} = 1$, $\Delta S'_{10,3} = 1$, $\Delta S'_{11,3} = 1$, and $\Delta S'_{12,3} = 1$, implying 14 S-boxes - $S1$, $S2$, $S3$, $S4$, $S14$, $S15$, $S16$, $S6$, $S7$, $S8$, $S9$, $S10$, $S11$, and $S12$ - in the next round are involved in the characteristic;

- when $\Delta S_{1,4} = 1$ and $\Delta S_{9,4} = 1$ are used in a characteristic, then $\Delta S'_{1,4} = 1, \Delta S'_{2,4} = 1, \Delta S'_{3,4} = 1, \Delta S'_{4,4} = 1, \Delta S'_{14,4} = 1, \Delta S'_{15,4} = 1, \Delta S'_{16,4} = 1$, $\Delta S'_{6,4} = 1$, $\Delta S'_{7,4} = 1$, $\Delta S'_{8,4} = 1$, $\Delta S'_{9,4} = 1$, $\Delta S'_{10,4} = 1$, $\Delta S'_{11,4} = 1$, and $\Delta S'_{12,4} = 1$, implying

14 S-boxes - $S1$, $S2$, $S3$, $S4$, $S14$, $S15$, $S16$, $S6$, $S7$, $S8$, $S9$, $S10$, $S11$, and $S12$ - in the next round are involved in the characteristic.

Hence when S-boxes $S1$ and $S9$ have 2 or more outputs of the same position involved in a characteristic, the number of S-boxes involved in the next round must be at least 14.

Based on these arguments and Lemma 5.6, Lemma 5.7 is proven for this case.

It is not hard to show that Cases 2.3, 2.4, and 2.6 follow the same argument as case 2.1, and case 2.5 follows the argument of case 2.2. Hence, the lemma is proven. □

Lemma 5.8 *If 2 S-boxes in round r are involved in a characteristic and they cause 4 S-boxes in the next round to be involved, then each of the 4 S-boxes in round $r+1$ must only have one input bit involved in the characteristic.*

Proof. Suppose S-boxes Si and Sj in a round are involved in a characteristic. As the number of involved S-boxes in the next round is 4, by Lemma 5.6, $\Delta S_{i,k} = 1$ and $\Delta S_{j,k} = 1$, where $k \in \{1, 2, 3, 4\}$, must appear simultaneously. Moreover, by the definition of the linear transformation, when $S_{i,k}$ and $S_{j,k}$ use a different Q_k^{mn} defined in the linear transformation, where mn denotes a combined partition number, $\Delta S_{i,k} = 1$ and $\Delta S_{j,k} = 1$ will respectively cause 7 bits that belong to 7 different S-boxes to be changed, and the 7 bits changed by $\Delta S_{i,k} = 1$ can not be cancelled by the bits changed by either $\Delta S_{i,l} = 1$ or $\Delta S_{j,l} = 1$, where $l \neq k$. Thus when the number of involved S-boxes in the next round is 4, $\Delta S_{i,k} = 1$ and $\Delta S_{j,k} = 1$ must appear simultaneously, and $S_{i,k}$ and $S_{j,k}$ must be involved in the same Q_k^{mn}.

When $S_{i,k}$ and $S_{j,k}$ are involved in the same Q_k^{mn}, and $\Delta S_{i,k} = 1$ and $\Delta S_{j,k} = 1$ are simultaneously satisfied, $\Delta Q_k^{mn} = 1 \oplus 1 \oplus 0 \oplus 0 \oplus 0 \oplus 0 \oplus 0 \oplus 0 = 0$. Thus $\Delta S_{i,k} = 1$ and $\Delta S_{j,k} = 1$ will cause only 2 bits which belong to 2 different S-boxes to be changed.

On the other hand, by the definition of the linear transformation, if $\Delta S_{i,k} = 1$ and $\Delta S_{j,k} = 1$ cause 2 bits that belong to 2 different S-boxes to be changed, when k is varied, the 2 bits changed will correspondingly belong to 2 other S-boxes.

It is assumed that the number of involved S-boxes in the next round is 4. Based on the above reasons, this case happens when 2 pairs of $\Delta S_{i,k} = 1$ and $\Delta S_{j,k} = 1$ occur, and in this case

each of the 4 involved S-boxes in the next round has one input involved in the differential characteristic. □

Lemma 5.9 *If 4 S-boxes in one round are involved in a characteristic, and at least one of them have 2 or more inputs involved in the characteristic, then the number of involved S-boxes in the previous round can not be 2.*

Proof. We prove the lemma by contradiction. Suppose the number of involved S-boxes in the previous round is 2. By Lemma 5.8, each of the 4 involved S-boxes in this round must have only one input involved in the characteristic. This is contradicting to the premise in the lemma that at least one of 4 S-boxes has 2 or more inputs involved in the characteristic. □

Until now we have proved the lemmas with regard to the number of involved S-boxes in the next round. By noticing Lemma 4.2 in Chapter 4 which states that the inverse linear transformation is exactly the same as the original one, all the arguments for the proof of the number of S-boxes in the next round can be applied to the number of S-boxes in the previous round. Thus the number of involved S-boxes in the previous round satisfies the same rules as that in the next round.

Theorem 5.1 *When an SPN consists of the linear transformation of Table 4.1 and the S-boxes satisfying diffusion order $\lambda = 1$, the average number of S-boxes involved in a one-round characteristic is 3 (except for the first and the last round).*

Proof. We shall prove the theorem by considering 3 cases.

Case 1: Suppose one S-box in one round is involved in a characteristic.

According to Lemma 5.5, the number of involved S-boxes in either the previous or next round must be at least 7. Hence we only need to consider 2 adjacent rounds. Since there are at least $(1 + 7)$ involved S-boxes in 2 rounds, the average number for a one-round characteristic is $(1 + 7)/2 = 4$. Hence, in this case the theorem is true.

Case 2: Suppose 2 S-boxes in one round are involved in a differential characteristic.

Since the S-boxes satisfy diffusion order of 1, at least either ≥ 2 inputs or ≥ 2 outputs of an S-box are involved in a characteristic. By Lemma 5.7, either in the previous round or in the

next round the number of involved S-boxes is 4, 6, or not less than 7 (except if this round is the first or the last round).

Moreover, by Lemma 5.9, when 4 S-boxes in one round are involved in a characteristic, the number of involved S-boxes in the next and previous round can not be both 2 simultaneously.

Hence when 2 S-boxes in one round are involved in a characteristic, the smallest value of average number of S-boxes involved in a one-round characteristic is $(2+4)/2 = 3$.

Case 3: Suppose 3 or more S-boxes in one round are involved in a characteristic.

In this case the average number of involved S-boxes contributed by this round is at least 3.

When the average number of S-boxes involved in a one-round characteristic is computed, the above 3 cases should all be considered, and the theorem follows. □

5.5 Effectiveness in Thwarting Differential Cryptanalysis

In selecting 4×4 S-boxes to construct an SPN, S-boxes that satisfy (1) the nonlinearity property of equation (4.2), (2) for the differential property, diffusion order of $\lambda = 1$, and (3) the maximum S-box XOR pair probability $p_\delta = 4/16 = 1/4$, can be found by random searching.

Suppose a cryptanalyst uses differential cryptanalysis [3] to attack an 12-round SPN that employs the linear transformation and the specified kind of 4×4 S-boxes. According to formula (5.1) and Theorem 5.1,

$$N_D \geq \frac{1}{(p_\delta)^{(R-1)\cdot n_a}} = \frac{1}{(1/4)^{[(12-2)-1]\cdot 3+2\times 2}} = 2^{62} \tag{5.3}$$

where 12 refers to the number of rounds of an SPN, $(12-1)-2$ refers to the number of rounds in an $(R-1)$-round characteristic minus the first and last rounds, the first and last rounds are assumed to have 2 S-boxes involved in characteristic because of Lemma 5.5, and 3 is the lower bound for the average number of S-boxes involved in a one-round characteristic for the remaining rounds.

An 12-round SPN that utilizes the linear transformation has 2^{62} chosen plaintext-ciphertext pairs required for differential cryptanalysis. For a block size of 64, this is clearly a reasonable level of complexity to declare the cipher secure. For comparison, a cipher which uses a permu-

tation such as Ayoub's permutation of Figure 4.1 with S-boxes that do not satisfy $\lambda = 1$, as few as 2^{24} chosen plaintext/ciphertext pairs are required; if the S-boxes satisfy $\lambda = 1$, the SPN has the maximum lower bound of 2^{48}. Therefore the linear transformation of Table 4.1 effectively improves an SPN's resistance to differential cryptanalysis.

5.6 Conclusion

In this chapter we analysed the strength of the novel linear transformation against differential cryptanalysis. We proved that when an SPN is constructed from the linear transformation and 4×4 S-boxes that satisfy diffusion order of 1, the average number of S-boxes involved in a one-round characteristic is at least 3. By utilizing the linear transformation, a 12-round SPN has 2^{62} chosen plaintext-ciphertext pairs required for differential cryptanalysis.

From the results in this and the previous chapters, it is demonstrated that the linear transformation has the advantage in increasing the resistance of an SPN to thwart both linear cryptanalysis and differential cryptanalysis.

Chapter 6

Implementation of an SPN using an FPGA

In previous chapters, new S-box selection criteria and a linear transformation for S-box interconnection are proposed. To check the complexity of the digital hardware implementation of the SPN constructed from this kind of S-box and linear transformation, this chapter deals with the implementation of our SPN using a Field Programmable Gate Array (FPGA). The FPGA product selected for the implementation is a Xilinx logic cell array.

6.1 Background

In this section, we give some basic knowledge about the Xilinx logic cell array and VHSIC Hardware Description Language (VHDL) used to specify the design.

6.1.1 Xilinx Logic Cell Array

The field-programmable gate array (FPGA)[1] is a relatively new type of digital component for the construction of electronic systems. Many FPGA chips are prefabricated as an array of identical functional blocks along with an interconnection network, and as the name implies, their functionality can be configured in the field, that is, at the point of application. The particular function of each block and the connections between blocks are programmed by the user.

[1]This subsection is based on reference [21]

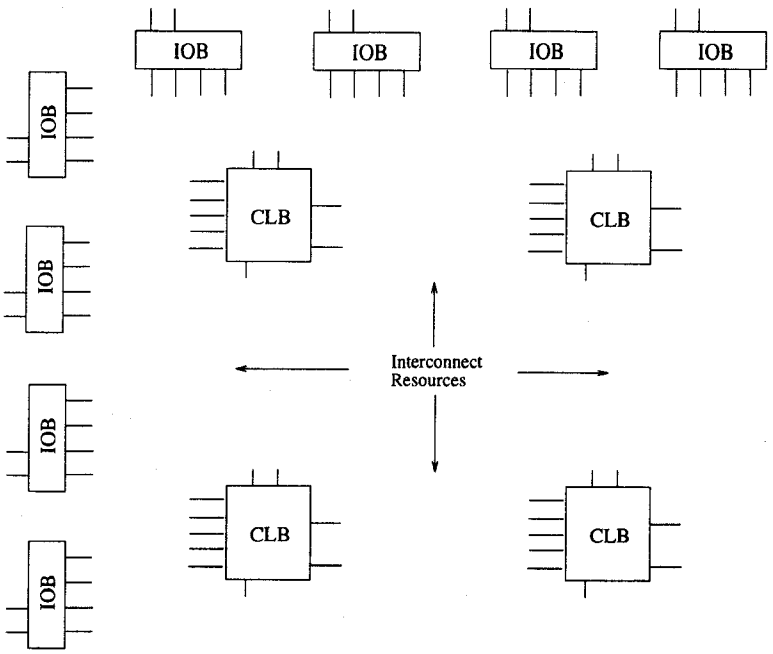


Figure 6.1: The Structure of Xilinx Logic Cell Array

Xilinx's proprietary logic cell array (LCA) architecture [29] is similar to that of other gate arrays, with an interior array of configurable logic blocks and a perimeter of input/output blocks. Horizontal and vertical routing channels run between the rows and columns of the logic blocks, and between the logic blocks and input/output blocks.

The programming method for LCA is based on CMOS static RAM technology: the function of logic blocks and the interconnection of signal paths are decided by the RAM cells that are scattered over the entire chip. The RAM cells linked together form a long shift register, and the programming is done by shifting in strings of ones and zeroes to configure the function of the chip. The configuration program is loaded automatically from an external memory on power-up or on command, or is programmed by a microprocessor as a part of system initialization [29].

In the design of the SPN, we target the XC4000 FPGA devices. The XC4000 series of programmable gate arrays is Xilinx's third generation static-memory-based FPGA architecture. As with the earlier XC2000 and XC3000 families, as shown in Figure 6.1 [14], the structure is based on three major configurable components: an array of configurable logic blocks (CLBs), a surrounding ring of input/output blocks (IOBs), and programmable interconnect resources.

The core of the device is a matrix of identical configurable logic blocks embedded in routing

resources. Figure 6.2 [21] is a block diagram of the principal elements within the XC4000 CLB. Each CLB includes three combinational function generators, two flip-flops, and their interconnect logic. Thirteen CLB inputs and four CLB outputs connect the function generators and flip-flops with the programmable interconnect lines surrounding the block. Four separate signals are supplied to each of two lookup-table-based function generators (F′ and G′). A third function generator (H′) can realize any Boolean function of its three inputs: the function F′ and G′ and a third input from outside the block (H1).

The two storage elements in the CLB are edge-triggered D-type flip-flops with common clock (K) and clock enable (EC) inputs, a third common input (S/R) that can be programmed as either an asynchronous set or reset signal, and programmable clock polarity.

The flexibility and symmetry of the CLB structure is advantageous for the placement and routing of a given application. Inputs, outputs, and the functions themselves can arbitrarily exchange positions within a CLB during placement and routing operations.

The perimeter of the Logic Cell Array is constituted by user programmable Input/Output Blocks. IOBs establish the connections between external package pins and the internal logic (see Figure 6.3 [21]). Each IOB controls one package pin and can be specified for input, output, or bi-directional signals.

An input signal can be routed to an input register that can be configured as either an edge-triggered flip-flop or a level-sensitive transparent latch. The optional delay on the data input to the register is used to compensate for the delay on a clock signal that first runs through a global buffer before reaching the IOB, without requiring any hold time on the data at the external pin.

An output signal can go directly to the pin or be registered in an edge-triggered flip-flop. The programmable output buffer controlled by an output enable signal allows three-state outputs or bi-directional pins. The output (O) and output enable (OE) signals are invertible, and the slew rate control is used to minimize power bus transients when switching non-critical signals.

Pull-up and pull-down resistors can be programmed to tie unused pins to Vcc or ground to probihit unnecessary power consumption. The clocks to the input and output registers are

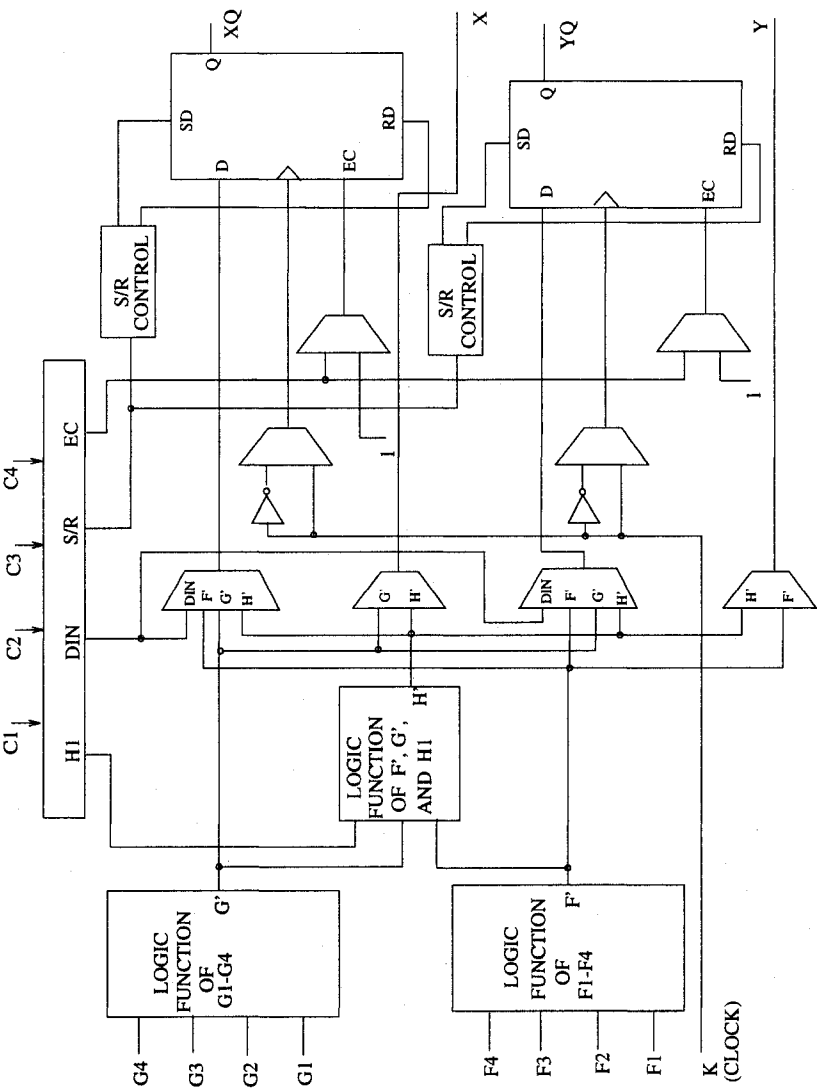


Figure 6.2: Configuration Logic Block

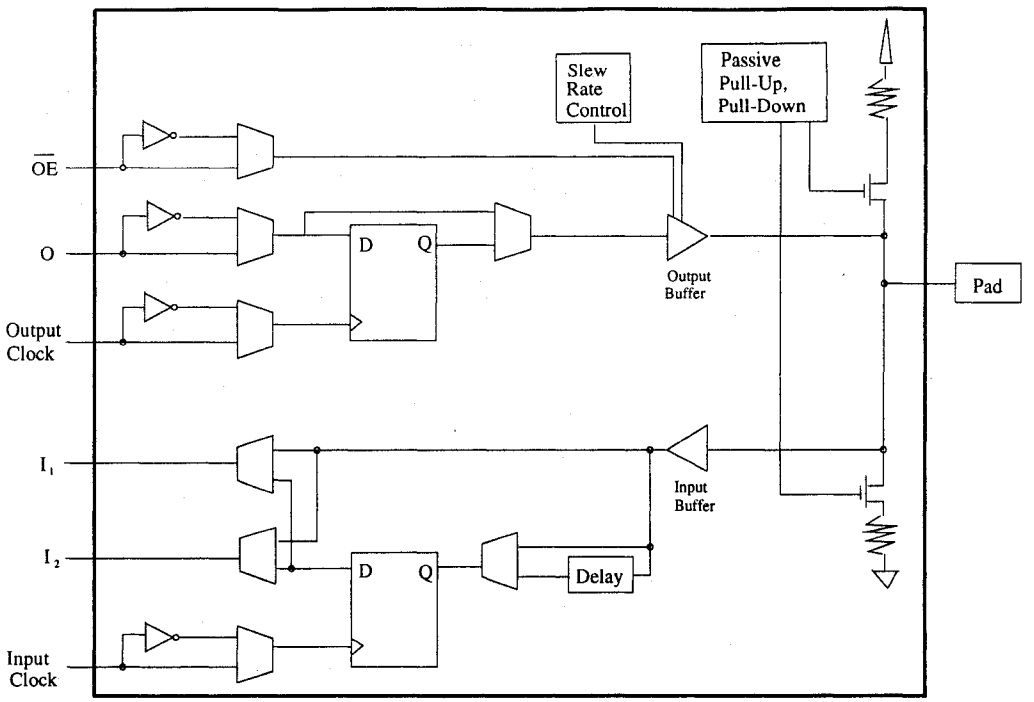


Figure 6.3: Input/Output Block

separate, and can be inverted, allowing either falling-edge or rising-edge triggered flip-flops. As with the CLB's registers, the input and output registers can be set or clear whenever the global RESET net is active.

The flexibility of the LCA is due to programmable routing resources that permit the interconnect of any two points on the chip. All internal connections are based on metal segments and programmable switching points. Programmable switch matrices implement the necessary connections between selected metal segments and block pins. There are three main types of interconnect, distinguished by the relative length of their segments: single-length lines, double-length lines, and long lines.

The single-length lines are a grid of horizontal and vertical lines that cross at a "switch matrix" between each block. Double-length lines bypass two CLBs before entering a switch matrix, providing efficient implementation of intermediate length interconnections. Long lines pass the entire breadth or length of the chip, and are intended primarily for high fan-out control signals. With a programmable "splitter switch" at its center, each vertical long line can be used as two separate routing channels that each run half the height of the chip. This hierarchy of

interconnection resources facilitates an efficient implementation of a given application.

6.1.2 VHDL

VHDL is an extensively used language for hardware description, built on the programming language ADA. It is used as the method to provide the input description for a number of commerically available computer-aided design systems.

A VHDL design entity (component, circuit, or system) consists of an external part (entity name and interface) and an internal part (entity implementation). After the external interface to an entity is specified, that entity can be used by other entities in a design. This concept of internal and external views is the core of a VHDL view of system design.

An entity is determined, with respect to other entities, by its interface and implementation. Several implementations or architectures can exist for one entity. Alternate architectures of an entity can be selectively used in a design without changing the rest of the design. An entity defined in a design can be reused in other designs, and libraries of entities can be developed for use by the entities of many designs.

A VHDL design entity has defined input, output, or input/output ports that are wired to neighbouring systems. An entity itself consists of interconnected entities, processes, and existing components, all which perform their tasks concurrently. Each entity architecture defines one implementation of the entity's function. An architecture is described by VHDL constructs such as arithmetic, signal assignment, or component instantiation statements. In VHDL, independent processes model sequential systems, such as counters, and combinational systems, such as AND or XOR gates. Processes can define and instantiate subdesigns. Processes communicate with the rest of the architecture by signals and port values. A signal has a source, one or more destinations, and a user-defined type.

A variety of constructs are used in VHDL to write design descriptions. With VHDL, digital systems of varying complexity (systems, boards, chips, modules) can be specified at varying levels of abstaction. VHDL language constructs can be split into categories by their level of abstraction. Three typical levels of abstractions are: behavioral, dataflow, and structural.

```
ENTITY count3 IS
        port ( clk, clr_bar : in std_logic;
                q_abc : out std_logic_vector ( 2 DOWNTO 0) ) ;
END count3;

ARCHITECTURE behavioural OF COUNT3 IS
        Signal internal_count : std_logic_vector ( 2 downto 0 ) := "000";
BEGIN
        counting : process ( clk, clr_bar)
                begin
                        if clr_bar = '0' THEN
                                internal_count <= "000" ;
                                        - - clear the counter;
                        elsif clk = '0' and clk'event THEN
                                internal_count <= internal_count + "001";
                                        - - keep counting;
                        end if;
        END process counting;
                q_abc <= internal_count;
END behavioural;
```

Figure 6.4: A VHDL description of a 3-bit counter

A behavioral description is the most abstract. It describes a design in an algorithmic form without caring about the detail as to how the design is to be implemented. A dataflow description models a design in the view of data flowing through the design from input to output. Operations are specified in terms of a set of data transformations, which are expressed as concurrent statements. A structural description is the most detailed. It defines a design with a list of components and their interconnections. A structural description is achieved by component instantiations right down to the gate-level.

VHDL itself is a large language, and learning all of it can be a very large task. In Figure 6.4, we give a sample behavioural description of a 3-bit counter, just to illustrate the kinds of capabilities VHDL provides. It is also worth to point out that only a subset of VHDL is synthesisable.

The entity declarations declare the external characteristics of a component - that is, the way it looks to the outside world. The architecture specifications define the internal operation. Note the similarity of the nature of a VHDL program to other programming languages.

6.1.3 Xilinx Synopsys Interface Program

In the implementation of our SPN using FPGA, the Xilinx Synopsys Interface (XSI^{TM}) program is used as the design automation environment.

The XSI design tool kit enables the user to use either the Synopsys FPGA Compiler or Design Compiler synthesis tool to implement Xilinx FPGA designs. The Synopsys FPGA Compiler and Design Compiler are High-level Design Automation (HLDA) tools. They create and optimize circuit designs from hardware descriptions written in VHDL or Verilog HDL.

For the Design Compiler, the following features are supplied [30]:

- Optimizes flip-flops and latches in the input/output block (IOB)
- Optimizes 3-state buffers in the IOB
- Enables one-hot state machines
- Uses the configurable logic block (CLB) Clock Enable pin automatically

For the FPGA Compiler, additional features are provided [30], such as:

- Optimizes logic to the XC4000 family CLB and IOB architectures
- Reports area and timing by device architecture, for example, CLB, IOB, and 3-state buffer

XSI supports both functional and timing simulation. The functional simulation may be used to debug the logic in a source design before implementing an FPGA. The timing simulation is used to verify the timing and functionality of the circuit after fitting a design into an FPGA.

By using VHDL and the XSI design tool kit, the Xilinx implementation flow for a design can be simply described as in Figure 6.5. The design process starts with an VHDL description of the desired circuit functions and ends with a BIT file, a binary file that contains the configuration data for the design, and an LCA file, which can be used for back-annotation and simulation, where DC-shell, Design Analyzer and XMake are programs residing in the XSI.

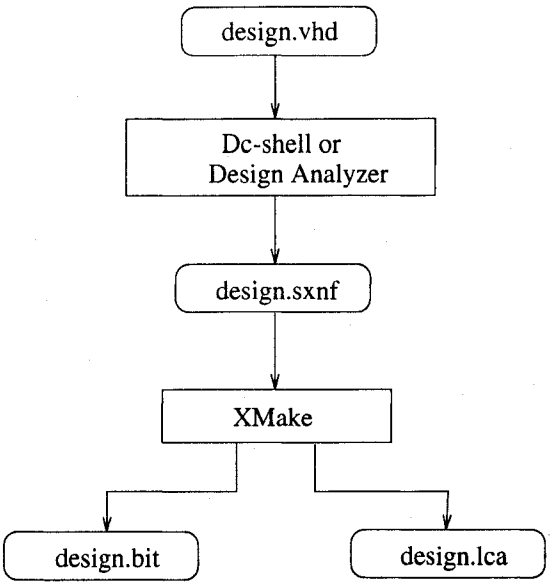


Figure 6.5: Design Flow Using VHDL

6.2 Architecture and Organization of SPN

The SPN algorithm is implemented as in Figure 6.6. It is a 12-round 64-bit block cipher with a 64-bit key, where each round consists of a layer of 16 4×4 S-boxes and a linear transformation. Keying the network is accomplished by XORing each bit of the 64-bit input to each round with a corresponding bit of the 64-bit cipher key and XORing each bit of the output from the last round with a corresponding bit of the same 64-bit key. Notice that we added a linear transformation in the last round to reduce the complexity of the circuit with no impact on security.

The 4×4 S-boxes used in the SPN satisfy two criteria: 1) For the differential property, the diffusion order of an S-box is 1; 2) For the the nonlinearity property, $2\eta(\gamma)$, defined in Chapter 3, satisfies (4.2).

Figure 6.7 shows the structure of the SPN. It consists of two main parts: a control unit and a data path. Five input signals and one output signal are used to control the operation of the control unit, and a 32-bit data bus is responsible for inputting/outputting data used for both data and keys from/to external devices. The operation of the logic control signals of Figure 6.7 will be outlined in Section 6.2.2.

The complete VHDL description of the SPN is given in [31].

6.2.1 Datapath

Figure 6.8 shows the main data paths of the SPN for encryption. The encryption process starts with the 64-bit key to be loaded into 64 D flip-flops by loading from 32-bit data inputs twice. With each load half of the key, key_1 or key_2, is clocked into 32 of the 64 D flip-flops dedicated to the key. After the key is loaded, since at this time the multiplexers Mux1 and Mux2 select data_1 and data_2 as inputs, respectively, the 64-bit data is written into 64 D flip-flops dedicated to data in two 32-bit loads. Once data_1 and data_2 are loaded into D flip-flops, Mux1 and Mux2 select feed_1 and feed_2 as inputs, respectively. Later on, after every one clock period, feed_1 and feed_2 are clocked into the corresponding D flip-flops. During each clock period, one round of SPN operation which consists of the substitution and linear transformation is performed, where S-boxes are implementated as 4 4-bit boolean functions. After feed_1 and feed_2 are clocked into the D flip-flops 12 times, the encryption operation of 12-round SPN is completed. The 64-bit ciphertext, which includes the 32-bit output_1 and the 32-bit output_2, is then read onto the data bus in 2 32-bit reads. The encryption is then complete.

In our design, either decryption and encryption can be done in the same chip. The overall data paths of the SPN are organized as in Figure 6.9. The signal Mode determines the working mode of the design. When Mode="1", the chip enters encryption mode. While Mode="0", the chip enters decryption mode. The decryption process is similar to that of encryption. The difference is the selection of S-boxes and the operational order of S-boxes and linear transformation. The S-box mappings used in the design are listed in Appendix A.

6.2.2 Control Unit Design of SPN

The diagram of control unit is depicted in Figure 6.10. At the left side of the diagram are the inputs to the control unit, and at the right side are the names of the controlled components which reside in the data path part of the SPN.

The control unit is designed to operate in this way: When the chip begins encryption or decryption, Clr_bar must be set low for one fourth Clock signal to clear all the counters. Then

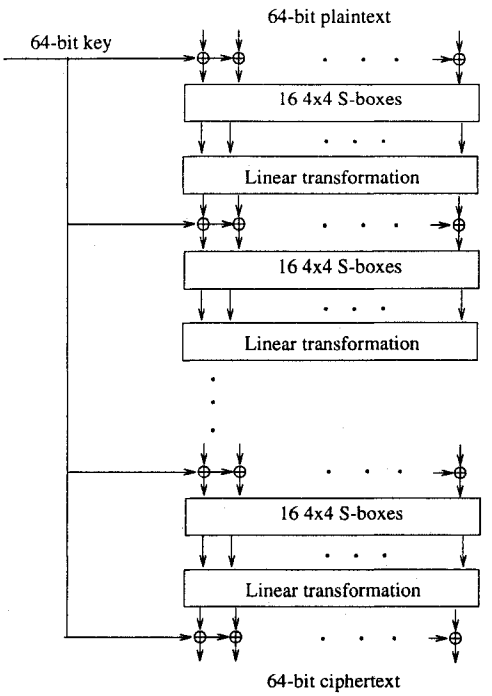


Figure 6.6: SPN Algorithm when Implemented

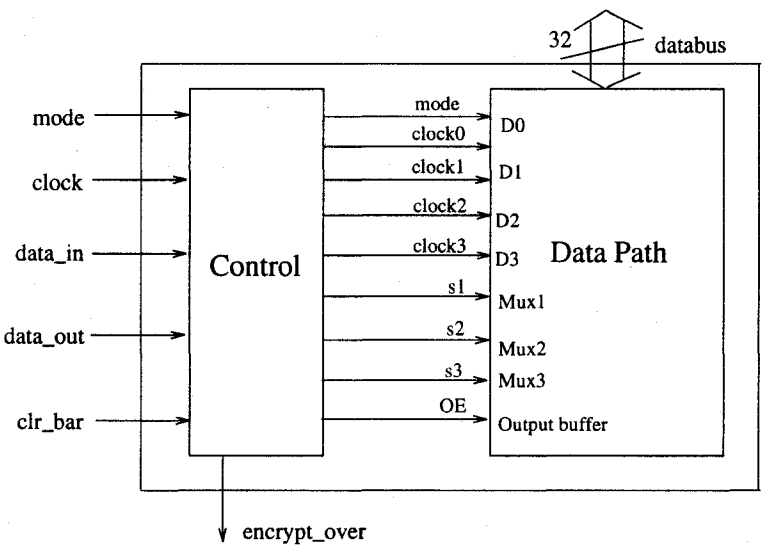


Figure 6.7: SPN Organization

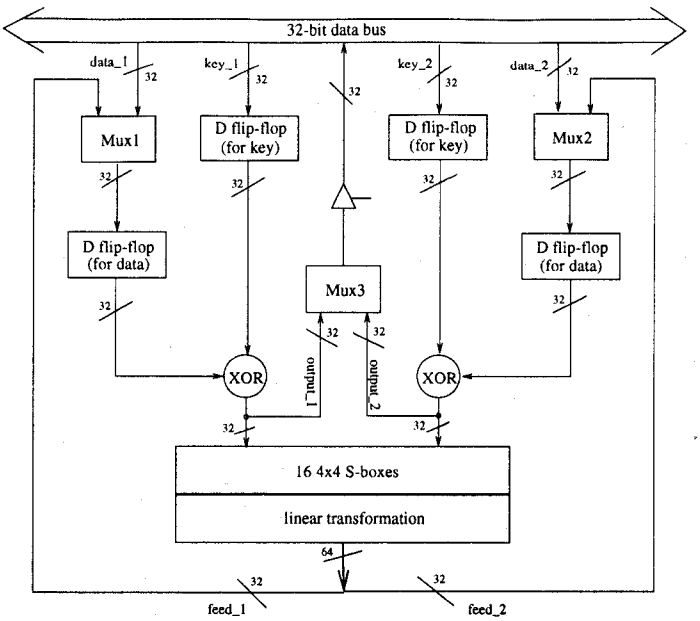


Figure 6.8: Data Paths for Encryption

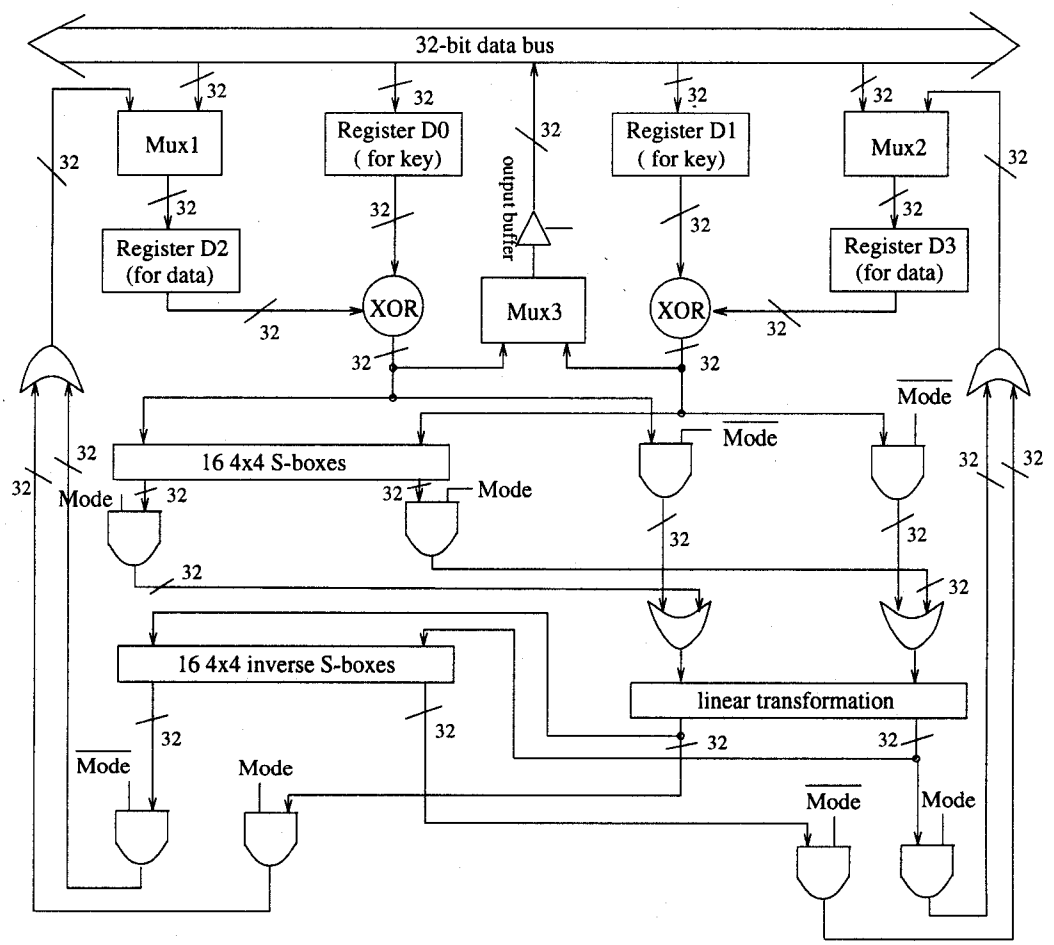


Figure 6.9: Detailed Data paths of the SPN

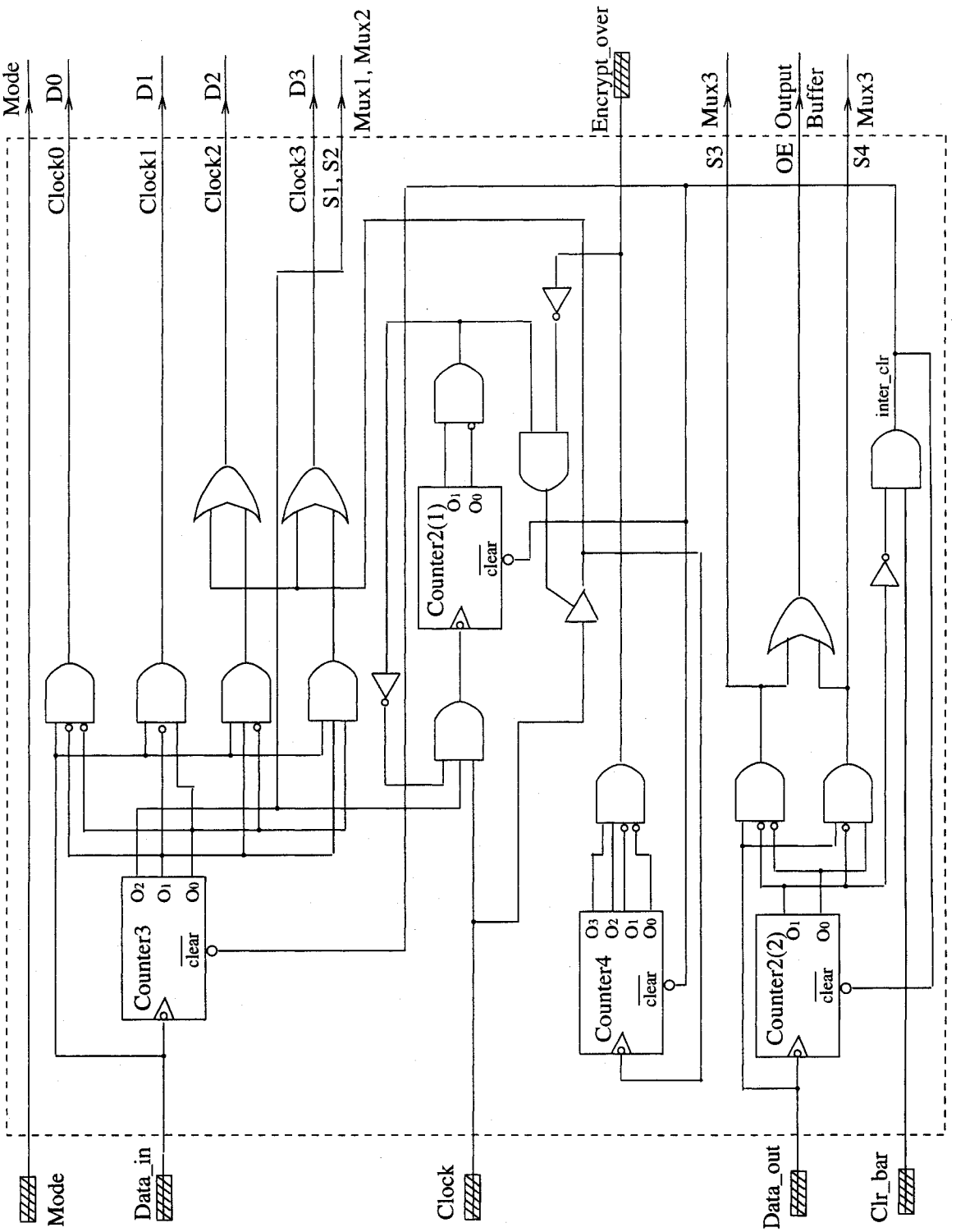


Figure 6.10: Control Unit of SPN

Data_in is given 4 pulses to generate one pulse for each of Clock0, Clock1, Clock2, and Clock3, successively. Thus the key bits, including key_1 and key_2, and the data bits, including data_1 and data_2, are clocked into the 32-bit registers of D0, D1, D2, and D3, respectively. After 4 pulses issued on Data_in, pin O_2 of Counter3 is high, letting Mux1 and Mux2 select feed_1 and feed_2 as inputs, respectively. Counter2(1) is used to guarantee that it is at least 1 Clock cycle after data_2 is clocked into D3 before Clock can connect to Clock2 and Clock3.

Once Clock connects to Clock2 and Clock3, after every Clock period feed_1 and feed_2 are clocked into D2 and D3 respectively. The SPN is constructed to have 12 rounds. Thus when Clock has generated 12 pulses on Clock2 and Clock3 as counted by Counter4, Clock is disconnected from Clock2 and Clock3, and at the same time signal Encrypt_over is set to high.

When Encrypt_over becomes high, Data_out is given 2 pulses by the external circuitry. During the first pulse, Mux3 selects output_1 as the input, and the output buffer is enabled to connect to the data bus. Thus the first half of the 64-bit ciphertext is sent out. For the second pulse, Mux3 selects output_2 as the input and the second half of the 64-bit ciphertext is sent out on the data bus. One complete cycle of encryption is then finished.

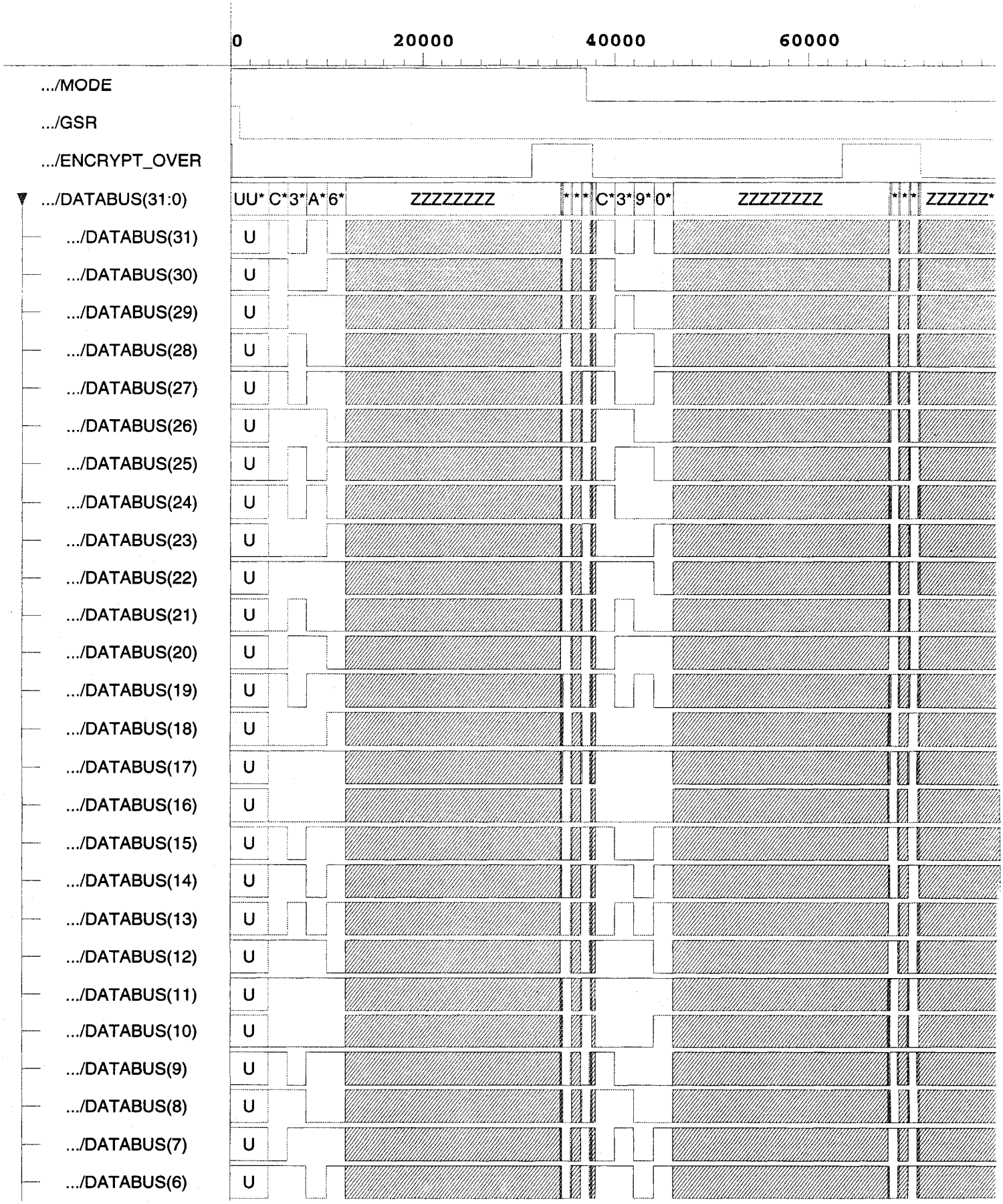
It is worthwhile to point out that, after Data_out is given 2 pulses, pin O_1 of Counter2(2) is high, and inter_clr is low. Hence all the counters will be reset, in preparation for the next encryption.

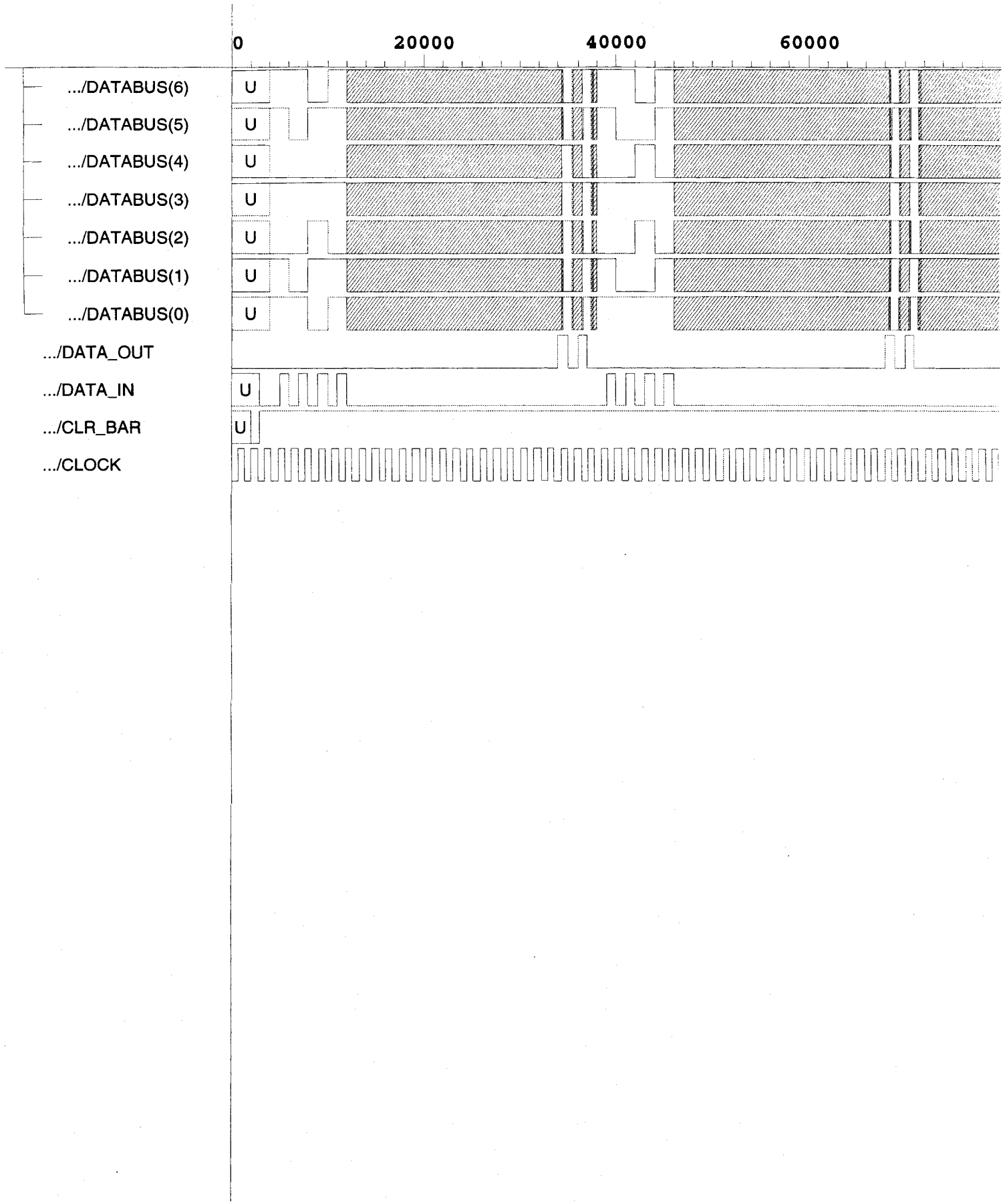
The encryption time can be calculated in this way. Let $T_{encryption}$ represent the total time for one encryption operation, t_{data_in} represent the length of one pulse on data_in, and t_{clock} represent the length of one Clock cycle, then $12.5t_{clock} + 4t_{data\_in} < T_{encryption} < 13.5t_{clock} + 4t_{data\_in}$, with no consideration of the propagation delays.

6.3 Simulation Results

A design should be tested by downloading the configuration bitstream of the design's BIT file into an FPGA chip. Since our intent was to just verify the concept, our design is verified by doing a detailed timing simulation based on Xilinx provided timing paramaters.

The timing simulation report of the design is given in the graph of the following two pages. As


0
20000
40000
60000
.../MODE
.../GSR
.../ENCRYPT_OVER
.../DATABUS(31:0)
UU* C*3*A*6*
ZZZZZZZZ
***
C*3*9*0*
ZZZZZZZZ
***
ZZZZZZ*
.../DATABUS(31)
.../DATABUS(30)
.../DATABUS(29)
.../DATABUS(28)
.../DATABUS(27)
.../DATABUS(26)
.../DATABUS(25)
.../DATABUS(24)
.../DATABUS(23)
.../DATABUS(22)
.../DATABUS(21)
.../DATABUS(20)
.../DATABUS(19)
.../DATABUS(18)
.../DATABUS(17)
.../DATABUS(16)
.../DATABUS(15)
.../DATABUS(14)
.../DATABUS(13)
.../DATABUS(12)
.../DATABUS(11)
.../DATABUS(10)
.../DATABUS(9)
.../DATABUS(8)
.../DATABUS(7)
.../DATABUS(6)
U


0
20000
40000
60000
.../DATABUS(6)
.../DATABUS(5)
.../DATABUS(4)
.../DATABUS(3)
.../DATABUS(2)
.../DATABUS(1)
.../DATABUS(0)
.../DATA_OUT
.../DATA_IN
.../CLR_BAR
.../CLOCK
U

can be seen from the waveforms, in this example encryption and decryption are both tested. At first, key CD4ADB6B367279C9 is loaded in and the plaintext AD5A9AAE6ACEEAEB is loaded and encrypted, yielding the ciphertext 925A181D0892ECEB. At the second time, the key CD4ADB6B367279C9 is loaded and the ciphertext 925A181D0892ECEB is decrypted, obtaining the plaintext AD5A9AAE6ACEEAEB. Obviously, encryption and decryption both are performed correctly. Also, actually different circuit modules are involved in encryption and decryption, it is extremely unlikely that some faults in the circuit of the SPN can be cancelled in the encryption and decryption. Hence the functionality of the design is illustrated and verified.

The encryption speed is mainly determined by how fast the circuit can be clocked by Clock. Based on our tests, at a clock cycle of $100ns$ for Clock, data can be encrypted and decrypted correctly, and $100ns$ is enough for the length of one data_in pulse. Thus, one encryption time is about $T_{encryption} = 13t_{clock} + 4t_{data\_in} = 13 \times 100 + 4 \times 100 = 1700ns$, and the encryption rate is $64 \times \frac{1}{1.7\mu} = 37.6Mbps$.

6.4 Complexity of the Design

The information given about the design is extracted from the report file "chip.rpt", which is generated by running the XMake command. Although the number of used CLBs is close to limit, according to the used numbers of the F, G, and H function generators, we can see that the structure of our SPN is not complicated. Also, since the design just occupied a small portion of the provided pins, it could have been designed with 64-bit I/O.

6.5 Conclusion

By using VHDL, our substitution-permutation network is demonstrated that the SPN can be easily implemented. We have verified that the design can be fitted into an XC4013PQ160-5 device. Though there are some drawbacks with the current implementation, e.g., (1) the same key is applied to every round, and (2) key loading is needed for every encryption or decryption. However, since the purpose of our hardware design and simulation is "proof-of-concept" in nature, these minor deficiencies are not of concern. It is likely with VLSI technology our SPN algorithm can be implemented more efficiently as part of a larger circuit. Hence it is realistic

PPR RESULTS FOR DESIGN CHIP

	No. Used	Max Available	%Used
Occupied CLBs	541	576	93%
Bonded I/O Pins	38	129	29%
F and G Function Generators (*)	730	1152	63%
H Function Generators	157	576	27%
CLB Flip Flops	107	1152	9%
IOB Input Flip Flops	32	192	16%
IOB Output Flip Flops	0	192	0%
3-State Buffers	64	1248	5%
3-State Half Longlines	64	96	66%
Edge Decode Inputs	0	288	0%
Edge Decode Half Longlines	0	32	0%
CLB Fast Carry Logic	4	576	0%

to put our SPN in practical use.

Chapter 7

Conclusions

The target set for the design of a block cipher is that the cipher is not only cryptographically strong but also simply implemented in software, hardware or both. This thesis has presented work which strongly promotes the attainment of the target.

7.1 Summary of the Thesis

An SPN consists of a number of rounds of substitutions (S-boxes) which are connected by S-Box interconnection layers. As a block cipher, an SPN is also vulnerable to two powerful cryptanalysis techniques of block ciphers: linear cryptanalysis and differential cryptanalysis.

In linear cryptanalysis probable linear approximations of a cipher are used to determine key bits. The probable linear approximations of a cipher are obtained by exploiting the linear properties of the S-boxes and the structure of the cipher. To strengthen an SPN in thwarting linear cryptanalysis, the design of S-boxes is considered. A new S-box design criterion is consequently suggested by noticing the characteristic of the basic SPN structure implying that a larger-term linear approximation of an S-box causes more S-boxes to be involved in a probable linear approximation of a cipher. When a basic SPN is constructed from the S-boxes that satisfy the new criterion, the capacity of the SPN to resist linear cryptanalysis is enhanced.

To improve the resistance of a basic SPN to linear cryptanalysis, we investigated one approach which involves the rearrangement of the permutation for each round. Based on our computational experiments, the method is not effective and was rejected.

Small S-boxes are easier to implement in hardware than large S-boxes. After checking the linear properties of small S-boxes, 4×4 S-boxes were selected to be adopted in the SPN that would be implemented in hardware. The thesis then focuses on the investigation of this kind of SPN.

Previously proposed linear transformations provide no advantage in resisting linear cryptanalysis when used in an SPN with the 4×4 S-boxes that satisfy our new design criterion. A new linear transformation is proposed such that an SPN constructed from the linear transformation and the 4×4 S-boxes is remarkably improved in resisting linear cryptanalysis and differential cryptanalysis.

An important part of the thesis is the implementation of the SPN which consists of our new linear transformation and the 4×4 S-boxes satisfying our new design criterion. FPGAs are used to investigate the implementation of the SPN. The simulation results demonstrate that the digital hardware implementation of the SPN is practical and not complicated.

7.2 Future Work

To consider an SPN for practical use, some further research work should be pursued.

Key scheduling, one of the three components of an SPN, is not studied in the thesis. A simple but secure key scheduling algorithm needs to be put forward in future designs. Methods are needed to prove the security of a key scheduling algorithm.

For practical applications, an SPN constructed from 8×8 S-boxes appears only suitable to be implemented in software. The main reason is that 8×8 S-boxes are not complicated to be realized in software by lookup-table method but are difficult to be implemented in hardware by using Boolean functions. Our new criterion for the design of S-boxes greatly raises the nonlinearity of fewer-term probable linear approximations of an 8×8 S-box. The existing linear transformations can not realize fully the advantage of the 8×8 S-boxes. A corresponding new linear transformation may be investigated.

An SPN consisting of 4×4 S-boxes is intended to be implemented in hardware. Implementing an SPN using FPGA is the first step to check the complexity of the hardware implementation. To see the actual complexity of the hardware design of an SPN, a VLSI implementation could

also be investigated.

The goal of SPN designers is to put the SPN into practical use and even to replace DES with the SPN. The results presented in this thesis strongly expedite the achievement of the goal, although some further research work needs to be conducted.

Bibliography

[1] C. M. Adams and S. E. Tavares, "The structured design of cryptographically good S-boxes," *Journal of Cryptology*, 3(1): 27-41, 1990.

[2] F. Ayoub, "The design of complete encryption networks using cryptographically equivalent permutations," *Computers and Security*, 2:261-267, 1982.

[3] E. Biham and A.shamir, "Differential cryptanalysis of DES-like cryptosystems", *Journal of Cryptology*, vol.4, no.1, pp. 3-72, 1991.

[4] E. Biham and A. Shmir, "Differential cryptanalysis of FEAL and N-Hash," *Advances in Cryptology: Proceedings of EUROCRYPT'91*, Springer-Verlag, Berlin, pp. 1-16, 1991.

[5] L. O'Connor, "An analysis of product ciphers based on the properties of Boolean Functions," *PhD thesis*, University of Waterloo, Waterloo, Ontario, 1992.

[6] M. H. Dawson and S. E. Tavares, "An expanded set of S-box design criteria based on information theory and its relation to differential-like attacks," *Advances in Cryptology: Proceedings of Eurocrypt'91*, pp. 352-367, Springer-verlag, Berlin, 1991.

[7] Department of Commerce, National Institute of Standards and Technology, "Announcing development of a Federal information processing standard for advanced encryption standard," *published in the January 2, 1997 issue of the Federal Register.*

[8] D. L. Dietmeyer, "Logic design of digital systems ", *Allyn and Bacon, Inc*, 1978.

[9] H. Feistel, "Cryptography and computer privacy", *Scientific American*, 228, pp. 15-23, 1973.

[10] H. Feistel, W. A. Notz, and J. L. Smith, "Some cryptographic techniques for machine-to-machine data communications," *Proceedings of the IEEE*, vol. 63, n0. 11, pp. 1545-1554, 1975.

[11] R. Forré, "Methods and instruments for designing S-boxes," *Journal of Cryptology*, 2(3): 115-130, 1990.

[12] H. M. Heys and S. E. Tavares, "The design of substitution permutation network ciphers resistant to cryptanalysis", *Journal of Cryptology*, vol.9, no. 1, pp. 1-19, 1996.

[13] J. B. Kam and G. I. Davida, "Structured design of substitution-permutation encryption networks," *IEEE Transaction on Computers*, c-28, pp. 747-753, 1979.

[14] R. H. Katz, "Contemporary logic design ", *The Benjamin/Cummings Publishing Company, Inc.* , 1994.

[15] X. Lai and J. Massey, "A proposal for a new block encryption standard," *Advances in Cryptology: Proceedings of EUROCRYPT'90*, Springer-Verlag, pp. 389-404, 1991.

[16] M. Matsui, "Linear cryptanalysis method for DES Cipher", *Proceedings of Eurocrypt'93*, Springer-verlag, Berlin, pp. 386-397.

[17] National Bureau of Standards FIPS Publication 46, "Data Encryption Standard (DES)", 1977.

[18] Z. Navabi, "VHDL analysis and modelling of digital systems ", *McGraw-Hill, Inc.* 1993.

[19] K. Nyberg, "On the construction of highly nonlinear permutations," *Advances in Cryptology: Proceedings of EUROCRYPT'91*, Springer-verlag, Berlin, pp. 1-16, 1991.

[20] K. Nyberg, "Perfect nonlinear S-boxes," *Advances in cryptology: Proceedings of EUROCRYPT'91*, Springer-verlag, Berlin, pp. 378-386, 1991.

[21] J. V. Oldfield and R. C. Dorf, "Field programmable gate arrays", *John Wiley & Sons, Inc.*, 1995.

[22] R. L. Rivest, "The RC5 encryption algorithm," *Dr. Dobb's Journal*, Vol. 20, no.1, pp. 146-148, Jan. 1995.

[23] R. L. Rivest, "The RC5 encryption algorithm," *K. U. Leuven Workshop on Cryptographic Algorithms*, Springer-Verlag, 1995.

[24] C.E. Shannon, "Communication theory of secrecy systems", *Bell systems Technical Journal*, vol. 28, pp.656-715, 1949.

[25] A. Shimizu and S. Miyaguchi, "Fast data encipherment algorithm FEAL," *Transactions of IEICE of Japan*, vol. J70-D, no. 7, Jul 87, pp. 1413-1423. (In Japanese.)

[26] D. R. Stinson, "Cryptography: theory and practice," *CRC Press, Inc.* 1995.

[27] A. F. Webster and S. E. Tavares, "On the design of S-boxes," *Advances in Cryptology: Proceedings of CRYPTO'85*, pp. 523-534, Springer-Verlag, Berlin, 1985.

[28] M. J. Wiener, "Efficient DES key search," *presented at CRYPTO'93*, Santa Barbara, Calif., August 1993.

[29] Xilinx, "The programmable gate array data book", *Xilinx, Inc.*, 1989.

[30] Xilinx, "Synopsys(XSI) for FPGAs interface tutorial guide", *Xilinx, Inc.*, October, 1995.

[31] J. Xu, "VHDL code for the design of an advanced substitution-permutation encryption network," *Report*, Memorial University of Newfoundland, 1997.

[32] A. M. Youssef, S. E. Tavares, and H. M. Heys, "A new class of substitution-permutation networks," presented at *Workshop on Selected Areas in Cryptography (SAC'96)*, Kingston, Ont., Aug., 1996.

Appendix A
SUBSTITUTION BOX VALUES

Note: *In the implementation of the SPN, the following 16 S-boxes are used for encryption. The corresponding 16 inverse S-boxes used for decryption can be trivially obtained.*

Substitution Box # 1

Input:	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓
Output:	3	15	0	12	8	5	13	10	6	9	11	2	1	14	7	4

Substitution Box # 2

Input:	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓
Output:	10	5	3	8	15	12	9	7	13	0	4	14	1	6	2	11

Substitution Box # 3

Input:	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓
Output:	9	14	5	0	6	11	15	12	2	13	8	3	1	7	4	10

Substitution Box # 4

Input:	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓
Output:	2	13	11	0	7	4	1	15	5	8	12	6	9	14	10	3

Substitution Box # 5

Input:	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓
Output:	1	6	13	8	14	3	7	4	10	5	0	11	9	15	12	2

Substitution Box # 6

Input:	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓
Output:	11	8	7	13	14	5	1	6	12	2	9	4	0	15	10	3

Substitution Box # 7

Input:	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓
Output:	13	3	11	5	6	12	8	15	1	4	2	14	10	9	7	0

Substitution Box # 8

Input:	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15

	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓
Output:	10	15	12	3	9	0	7	13	4	2	1	8	14	5	11	6

Substitution Box # 9

Input:	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓
Output:	3	0	15	5	6	13	9	14	4	10	1	12	8	7	2	11

Substitution Box # 10

Input:	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓
Output:	5	11	3	13	14	4	0	7	9	12	10	6	2	1	15	8

Substitution Box # 11

Input:	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓
Output:	6	13	9	2	15	8	3	5	1	4	7	14	12	11	10	0

Substitution Box # 12

Input:	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓
Output:	14	5	1	10	7	0	11	13	9	12	15	6	4	3	2	8

Substitution Box # 13

Input:	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓
Output:	8	2	5	12	7	4	9	15	3	14	0	11	10	13	6	1

Substitution Box # 14

Input:	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓
Output:	0	13	9	14	15	6	3	8	5	11	10	7	2	12	4	1

Substitution Box # 15

Input:	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓
Output:	4	9	3	15	13	14	8	1	2	5	12	6	11	0	7	10

Substitution Box # 16

Input:	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓
Output:	7	8	10	1	9	3	6	15	0	13	12	2	5	14	11	4