DESIGN OF FPGA-BASED ENCRYPTION CHIP USING BLOWFISH ALGORITHM

By

Khor Lay Hoong

This dissertation is presented to

UNIVERSITI SAINS MALAYSIA

As partial fulfillment of requirement for the degree with honours

BACHELOR OF ENGINEERING (ELCTRONIC ENGINEERING)

School of Electrical and Electronic Engineering

Universiti Sains Malaysia **May 2006**

ABSTRACT

Nowadays, the world has changed so rapidly that everything has become digitized and computerized. Unfortunately, digital information is very easy to be duplicated, modified, transmitted or used by unauthorized users. This results a serious problem and in view of this, some sort of security mechanism has to be produced to protect it. This is where the study of cryptography comes in. Cryptography has been introduced to protect the information. However, until now, the cryptography hardware is still not commonly used especially in FPGA. In this project, the Blowfish encryption algorithm is chosen because it is among the safest algorithm used nowadays. The aim of this project is to design a Blowfish encryption chip in FPGA. For this project, the design entry used is Altera's Quartus II Version 5.0 and the targeted hardware is Altera's Flex10K FPGA device. By using FPGA device, data can be encrypted or decrypted in real time and this would be a great tool for security purpose, such as ATM machine. The first stage of this project is the study of Blowfish algorithm and translates the method into VHDL code because VHDL has been commonly used as a design entry language for FPGA in digital design. Producing the VHDL code is the most difficult and time-consuming part throughout this project. In the second stage, the design is realized using the FPGA board. In this stage, timing is the most critical factor that must be taken care of. If the timing is incorrect, the output may be wrong. Comparison will be done on the software result and hardware result to ensure that the encryption chip is designed correctly and function well.

ABSTRAK

REKABENTUK SUATU CIP INKRIPSI FPGA MENGGUNAKAN ALGORITMA BLOWFISH

Sejak kebelakangan ini, perkembangan teknologi elektronik adalah begitu pesat sehingga penggunaan sektor pengkomputeran menjadi semakin meluas. Malangnya, informasi komputer ini terlalu senang diciplak, disunting dan dicuri oleh pihak yang tidak bertanggungjawab. Memandangkan masalah ini sangat merumitkan, maka satu mekanisma keselamatan haruslah dicipta bagi mengatasi masalah ini. Mekanisma yang dimaksudkan ialah kriptografi. Sememangnya, sehingga kini, perkakasan kriptografi yang wujud masih terhad terutamamnya perkakasan FPGA. Dalam projek ini, algoritma kriptografi Blowfish dipilih kerana algoritma ini adalah antara algoritma yang paling selamat digunakan. Tujuan projek ini adalah untuk merekabentuk satu cip inkripsi menggunakan FPGA. Bagi projek ini, perisian yang digunakan ialah Altera's Quartus II Versi 5.0 manakala perkakasan yang digunakan ialah Altera's Flex10K peranti FPGA. Dengan menggunakan FPGA, data boleh diinkrip dan didikrip dalam keadaan yang benar dan ini boleh dijadikan sebagai perkakas keselamatan yang berguna, misalnya mesin ATM. Peringkat pertama bagi projek ini ialah kajian terhadap algoritma Blowfish dan selepas itu kod VHDL dihasilkan. Penghasilan kod VHDL paling menyusahkan dan memakan paling banyak masa sepanjang tempoh projek ini. Bagi peringkat kedua pula, rekabentuk ini akan diimplimentasikan ke atas perkakasan FPGA. Dalam implimentasi perkakasan, faktor yang paling kritikal is faktor masa dan harus diambil kira dengan sebaik-baiknya. Akhir sekali, perbandingan antara keputusan perisian dan keputusan perkakasan akan dibuat untuk memastikan cip inkripsi yang direkabentuk berfungsi dengan betul.

ACKNOWLEDGEMENT

Although this dissertation is an individual task, it would not have been possible without the support and help of many people who contributed in innumerable ways.

First and foremost, I would like to take this opportunity to express my deepest appreciation to my supervisor, Prof. Madya Dr. Othman Sidek and Pn. Dzati Athiar Ramli for their superb guidance and help rendered to me in order for me to complete this dissertation.

A token of appreciation is also dedicated to my parents and my siblings who always support and encourage me whenever I face any difficulties. With their support, I become more confident in bringing this dissertation to completion.

My gratitude also goes to Mr. Chew Hon Wah, Mr. Ng Soo Kheng and Mr. Lim Mui Liang for their suggestions and assistance. Not forgetting the DSP lab technicians and Universiti Sains Malaysia for providing resources and means to complete this project.

I am greatly in debt to all of those who mentioned above throughout the whole period of doing this project. Once again, thank you.

CONTENTS

FIGURE LIST

TABLE LIST

ABBREVIATION LIST

Abbreviaton	Explanation
FPGA	Field Programmable Gate Array
VHDL	Very High Speed Integrated Circuit Hardware Description Language
CPLD	Complex Programmable Logic Device
ASIC	Application Specific Integrated Circuit
SOPC	System-on-programmable-chip
LE	Logic Elements
RQFP	Power Quad Flat Pack
LUT	Look-up Table
RAM	Random Access Memory
ECC	Elliptic Curve Cryptosystems
DSP	Digital Signal Processing
OTP	One-time Programmable
AES	Advance Encryption Standard
DES	Data Encryption Standard
IDEA	International Data Encryption Algorithm
RSA	Riverst-Shamir-Adelman
SAFER	Secure and Fast Encryption Routine
RTL	Register Transfer Level
LED	Light Emitting Diode
FIFO	First-in First-out

CHAPTER 1
INTRODUCTION

1.1 Introduction

It is undeniable that in this modern world of technology, everything has become computerized. Nowadays, e-banking, e-commerce, digital signature, email, smart card, ATM card, e-registration and e-cash have become the popular trend of modern digital communication. However, what we worry the most is the security issue. Digital information is very easy to be duplicated, modified, transmitted or used by unauthorized users. This results a serious problem and in view of this, some sort of security mechanism has to be produced to protect against misuse by unauthorized persons. This is where the study of cryptography comes in.

1.2 Cryptography

Cryptography has been introduced to protect the information. In some applications, it is extremely essential to encrypt the information. For example, a patient's medical particular, saving account's password and so on. Cryptography involves the study, design and application of methods to ensure the security, privacy and integrity of information. (R.C.W. Phan, 2002)

1.2.1 Terminology of Cryptography

Cryptography was originated from 2 Greek words, which are *kryptós* and *gráphein. Kryptós* means "hidden" and *gráphein* means "to write" (Nichols, 1999). In short, cryptography means "secret writing".

1.2.2 History of Cryptography

Historically, cryptography was concerned solely with encryption, which is means of converting information from its normal, comprehensible form into an incomprehensible format and it is unreadable without secret knowledge. Generally, the earliest forms of secret writing was divided into two main categories, they are transposition ciphers, which rearrange the order of letters in a message, and substitution ciphers, which systematically replace letters or groups of letters with other letters or groups of letters. One of the earliest and simplest substitution ciphers was the Caesar cipher, used by Julius Caesar during his military campaigns. This shows that cryptography was used primarily for military communications. In recent decades, however, the field of cryptography has expanded its remit and has a variety of applications including authentication, digital signatures, electronic voting and digital cash. (Citing: http://en.www.wikipedia.com)

1.3 Encryption and Decryption

Encryption is the field concerned with linguistic and mathematical techniques for securing information, particularly in communications. In other words, that is a means of converting information from its normal, comprehensible form into an incomprehensible format, rendering it unreadable without secret knowledge. For brief understanding, encryption is the process of converting ordinary information (*"plaintext"*) into unreadable information ("*ciphertext"*). Of course, decryption is the reverse process of encryption, and will recover the plaintext back from the ciphertext. Encryption is done by the sender whereas decryption is done by the authorized recipient.

1.3.1 Process of Encryption

Figure 1.1 shows a basic encryption and decryption process. Let say, a sender, A, would like to send a message to his friend, B, which act as a receiver. In cryptography, the original message is call plaintext. A will use a key (a secret piece of information) to change the message into an incomprehensible format, namely ciphertext. Then, the encrypted text will be sent to B and B will open the message using a key so that the message becomes comprehensible (original message).

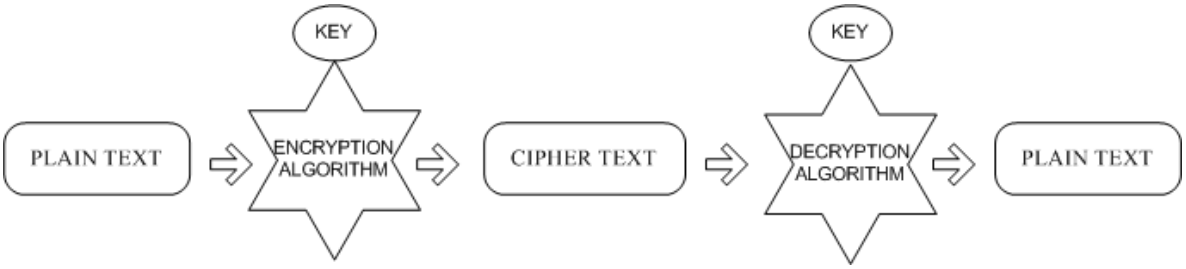

Figure 1.1: Graphical Representation of Encryption and Decryption Process

1.4 Objective and Scope of the Project

The main objective of this project is to design a FPGA-based encryption chip using Blowfish algorithm. The design is visualize using VHDL (Very High Speed Integrated Circuit Hardware Description Language) as design entry and it will be programmed into FPGA device to do the hardware implementation. Besides, this project also helps to promote the usage of VHDL and FPGA as a design medium for digital and logic design.

Basically, there are two main parts in Blowfish algorithm, a part that handles the expansion of the key and a part that handles the encryption of the data. All the sub keys needed to be generated first before dealing with the encryption part. This algorithm is then translated into VHDL coding so that the FPGA device can recognize it. Once the design in VHDL is done, Quartus II software is used to compile, verify, synthesize and simulate the design. After that, the design will be burned into the FPGA board so that the result can be shown physically.

1.5 Hardware and Software

The software used in this design is Altera's Quartus II Version 5.0. This design software is the most comprehensive environment available for system-on-programmable-chip (SOPC) design. Also, this software is very user-friendly, and has better capability. This software supports schematic capture and text-based hardware description language design entry, namely Verilog HDL, VHDL and AHDL$_{TM}$ (Altera Hardware Description Language). It also provides design programming, compilation, and verification support for all Altera's device including the EPF10K70 device which will be used in this project.

The FPGA board used in this project is Altera's University Program 2 Education Kit. The UP2 Education Kit was designed to meet the needs of universities teaching digital logic design with programmable logic devices (PLDs) The device used is Flex10K or known as EPF10K70. This device is build together in a 240-pin power quad flat pack (RQFP) package and has 3,744 logic elements and nine embedded array blocks. The EPF10K70 device is based on SRAM technology. Each logic elements consists of a four-input LUT, a programmable flip-flop, and a dedicated path for carry-and-cascade functions. Each embedded array block provides 2,048 bits of memory which can be used to create RAM, ROM, first-in first-out (FIFO) functions, implement state machines or multiplier. Besides, there are also 70,000 typical gates readily-available in the EPF10K70 device. (citing: http://www.altera.com)

1.6 Report Structure

The first chapter of this report is introduction. In this chapter, the importance, history and terminology of cryptography were clearly stated. Also, the basic knowledge regarding encryption and decryption process was illustrated. The final part in this chapter states the objective of this project and the tools needed to complete this design.

Chapter 2, "Literature Review" discuss about VHDL, FPGA and the advantages of using VHDL and FPGA. The Blowfish algorithm, which is the algorithm chosen to be used in this project, will be illustrated in this chapter too. Furthermore, the Blowfish algorithm will be compared with other encryption algorithm. Some famous product

which use Blowfish algorithm will be tabled in this chapter too. Finally, the main component in Blowfish algorithm such as the key, the sub-keys, the s-box, the p-array, and the Fiestel network will be explained in details.

Chapter 3, “Methodology” describes the steps to implement Blowfish algorithm. On the beginning part of this chapter, the mathematical equations and the theory part of Blowfish is mentioned with the help of flowcharts and some graphical representation. After that, the method of using the software and the hardware used will be explained, ranging from the steps to produce the design entry until connecting the hardware to the computer in order to download the program into the FPGA device. The last part of this chapter, which is also the core part, will explain the VHDL coding in details. The functionality of the design is explained part by part, including the method applied to solve the timing problem when dealing with the hardware.

Chapter 4, “Results” includes all of the compilation results, simulation results and timing analysis for this design. The simulation results are shown part by part and explanation for the particular simulation will come together with the simulation graph. In this chapter also, the results are proven by comparing with hand-calculated results. Finally, the simulation results of the full system will be shown. For this final part, two different keys (the minimum-length and the maximum-length key) are used to check the versatility of the design.

Chapter 5, “Conclusion” is the concluding chapter of this report. The first part of this chapter states the summary of the design while the second part of the chapter illustrates some of the suggestion made for future and further improvement regarding this project.

CHAPTER 2
LITERATURE REVIEW

2.1 Introduction

As mentioned in Chapter 1, cryptography and encryption has become an important field of study nowadays. Hence, many researchers have put their effort to go deeply into this field. Vincent Rijimen and Springer Verlag are among the famous cryptographers who did their Ph.D. thesis in Blowfish encryption algorithm.(Dr.Dobbs Journal, 1995). In Malaysia, Raphael Phan Chung Wei does a lot of cryptographic research. He also promotes cooperation and collaboration among Malaysian universities in pursuing cryptographic research. The main research in Malaysia is the development of a security mechanism in the national multi-purpose card or so called smart card. Besides, there are also many researchers studying in this field. For instant, UPM's Math Department has a team of researchers actively involved in this area, M.R.Said developed a public-key cryptosystem analogous to the LUC cryptosystem. The team also has interests in number theory and Elliptic Curve Cryptosystems (ECC). USM's School of Math has an Error Correcting Code and Public Key Cryptography research group under G.A.How that researches on the design of public key systems and RSA system. Meanwhile, MMU's Engineering Faculty has an Information Systems Security research group headed by M.U. Siddiqi that is involved in designing cryptographic systems. They designed a fair e-cash system for internet payment. (R.C.W. Phan, 2002).

This project also focuses on the encryption field. The algorithm used is Blowfish encryption algorithm. In this chapter, the literature review or the theory of FPGA, VHDL, encryption algorithm and Blowfish algorithm will be focused and discussed clearly. The advantages of FPGA and VHDL will be discussed in this chapter too.

2.2 FPGA

For this project, FPGA is used to implement the Blowfish encryption chip. FPGA, a field programmable gate array, is a semiconductor device used to process digital information, similar to a microprocessor. It can be reprogrammed after it is manufactured, rather than having its programming fixed during the manufacturing. In short, FPGA is a programmable logic device. The programmable logic components can be programmed to duplicate the functionality of basic logic gates, such as AND, OR, XOR, NOT or more complex combinational functions such as decoders, simple math functions or flip-flops. Nowadays, many works is done using FPGA approach. Applications of FPGA include DSP, aerospace, medical imaging, computer vision, speech recognition, cryptography, bioinformatics and so on. This is because the price of FPGA is affordable yet it can cater complicated digital design.

2.2.1 Types of FPGA

Basically, there are 2 types of FPGA, namely SRAM-based reprogrammable FPGA and one-time programmable FPGA which is also commonly known as OTP. SRAM-based reprogrammable FPGA is the dominant type of FPGA used. This type of FPGA can be reprogrammed by the user at any time. This means that user can design, program and make changes to his circuit whenever he wants. In fact, an SRAM FPGA is reprogrammed every time it is powered-up.

The FPGA used in this design is the SRAM-based reprogrammable FPGA. As for OTP, it is purposely meant for commercial used but not education-wise because once the program is downloaded to the board, it will make a permanent connection in the chip and the designer cannot make any changes to his program unless to change another chip.

2.2.2 Advantages of FPGA

A recent trend shows that FPGA is commonly used by designers to implement their complicated digital design. This is due to some notable advantages of the FPGA. Since there are many higher-level embedded functions (such as adders, multipliers, flip-flops and multiplexer) and embedded memories presence in most FPGA, a designers need not fabricate the chip themselves which consumes a lot of time and money. In other words, by using FPGA, the design made is cost-effective and time-effective.

Another advantage of FPGA is “fast”. By using FPGA, a designer can get the result in a very short time. Once the designer discover that the result is not correct, the FPGA can be easily reprogram and retest the result.

Also, the usage of FPGA can be further broadened by combining it with embedded microprocessors and related peripherals to form complete "systems on a programmable chip”.

2.3 VHDL

VHDL stands for VHSIC (Very High Speed Integrated Circuit) Hardware Description Language. In the mid 80’s, the U.S Department of Defense and the IEEE sponsored the development of this hardware description language with the goal to develop very high-speed integrated circuit. It has been commonly used as a design entry language for FPGA in electronic design automation of digital circuits, and is widely used in the industries. Nowadays, VHDL is becoming increasingly popular as a way to capture complex digital electronic circuits.

2.3.1 Types of Representation in VHDL

Basically, there are three types of representation in VHDL, namely behavioral, structural and data flow. This keeps the description and design of complex systems manageable. By using VHDL, a digital system can be represented at different levels of abstraction.

The first representation is behavioral that describe a system in terms of what it does or how it behaves rather than in terms of its components and interconnection between them. A behavioral description sometimes called as “black box” representation because it only specifies the relationship between the input and output signals. Figure 2.1 shows an example of the behavioral representation in VHDL.

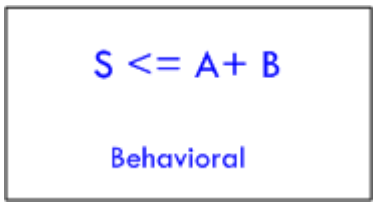

Figure 2.1: Behavioral Representation in VHDL

The structural level, on the other hand, describes a system as a collection of gates and components that are interconnected to perform a desired function. A structural description could be compared to a schematic of interconnected logic gates. It is a representation that is usually closer to the physical realization of a system. Figure 2.2 shows an example of the structural representation in VHDL.

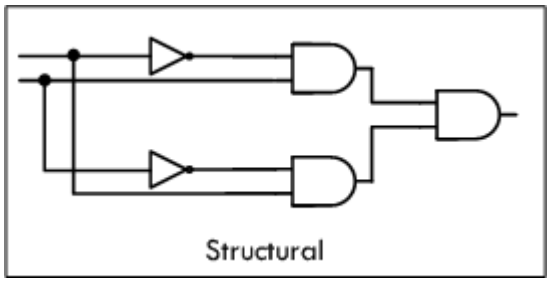

Figure 2.2: Structural Representation in VHDL

The dataflow representation describes how data moves through the system. This is typically done in terms of data flow between registers (Register Transfer level). The data flow model makes use of concurrent statements that are executed in parallel as soon as data arrives at the input.

2.3.2 Basic Terminology used in VHDL

In VHDL, there are some basic building blocks that are used in almost every design.

- Entity

  All designs are expressed in term of entities. An entity is the most basic building block in a design. The uppermost level of a design is the top level entity. If the design is hierarchical, then the top level description will have the lower level descriptions contained in it. These lower level descriptions will be lower level entities contained in the top-level entity description.

- Architecture

  All entities that can be simulated have an architecture description. The architecture describes the behavior of the entity. A single entity can have multiple architectures. One architecture might be behavioral while another might be a structural description.

- Package

  A package is a collection of commonly used data types and subprograms used in a design. Think of a package as a tool box that contains tools used to build designs. (Douglas L. Perry, 2002)

- VHDL uses reserved keywords that cannot be used as signal names or identifiers. Keywords and user-defined identifiers are case insensitive. Lines with comments start with two adjacent hyphens (--) and will be ignored by the compiler. VHDL also ignores line breaks and extra spaces.

2.3.3 Advantages of VHDL

Although VHDL look similar as conventional programming language, there are some important differences. The commands of a hardware description language (VHDL or Verilog) which correspond to logic gates are executed in parallel, as soon as a new input arrives. For such, VHDL can said to be a very powerful and versatile description language. Advantages of VHDL are stated below:

Digital circuits captured using VHDL can be easily simulated. VHDL's coding style is flexible and hence suitable for handling very complex designs. In other words, VHDL allows designers to model hardware at various level of design abstraction. VHDL can describe hardware as a "black box" which is just considering the relationship between input and output or design it using gate level description.

VHDL is a standard language, so there are many readily available design tools and designer is able to take advantage of the most up-to-date design tools. Since VHDL is a popular standard language, communication among designers is made easier.

VHDL support multiple levels of abstractions and it is possible to simulate designs in which we have high-level descriptions for some sub-modules with low-level implementations of other sub-modules. In short, a designer has many choices to present the idea in different style.

VHDL has versatile design reconfiguration support. Redesigns or alterations of a subsystem can be easily verified by just replacing the VHDL model for that sub-module and re-simulating with the original test set. A designer may reuse some of the common elements, packages and libraries in the previous design.

2.4 Encryption Algorithm

Currently, there are many encryption algorithms available, such as AES, DES, IDEA, KHUFU, RS6 and so on. The encryption algorithm used in this project is the Blowfish algorithm. The Blowfish algorithm was first introduced in 1993 by Bruce Schneier, one of the world's leading cryptologists. He is the president of Counterpane Systems (a consulting firm specializing in cryptography and computer security). Aside from his duties of presiding over Counterpane Systems, he has contributed many written documents on the subject of cryptography, including the book Applied Cryptography, *(John Wiley & Sons, 1994 & 1996)*, which has been described as an influential literary work in the field of cryptography.

An encryption algorithm is a mathematical procedure for performing encryption on data. Through the use of an algorithm, information is made into meaningless cipher text and requires the use of a key to transform the data back into its original form. Blowfish, AES, DES, SHARK, TEA, RC4, RC5 and RC6 are examples of encryption algorithms. Encryption algorithms can be divided into two main groups; they are symmetric key algorithms and asymmetric key algorithms.

2.4.1 Symmetric Key Algorithm

Another term for symmetric key algorithms is private-key cryptography. For this type of algorithms, the sender and the receiver use a pair of shared key which is kept secret from all other parties. This means that the sender and the receiver use the same key to perform encryption and decryption as well. Examples of encryption algorithm which use this type of key are DES, AES, IDEA, Twofish, RC4 and Blowfish. In symmetric key algorithm, the encryption key is closely related to the decryption key. The key might be identical or there is a simple transform to go between the two keys, for example in reverse order.

The encryption and decryption process of this algorithm can be illustrated using equation below:

$$E_K(M) = C \quad (2.1)$$

$$D_K(C) = M \quad (2.2)$$

Where E represents the encryption process, D represents the decryption process, M represents the plaintext, C represents the ciphertext and K represents the key.

Basically, symmetric key algorithms can be divided into two types, they are stream ciphers which encrypt the bits of the message one at a time, and block ciphers which encrypt a number of bits as a single unit. Blowfish is an algorithm which encrypts a block of 64 bits.

2.4.2 Asymmetric Key Algorithm

Another term for asymmetric key algorithms is public-key cryptography. Unlike the symmetric algorithm, asymmetric key algorithm uses two separate keys to perform encryption and decryption. It uses a public key and any sender can perform the encryption using this public key. To perform decryption, the receiver uses a private key which is kept secret. Example of encryption algorithms which use this type of key are RSA, SPEKE, DSA, Schnorr and so on. This technique enables a wide variety of applications, such as ATM machine, digital signature and so on.

The encryption and decryption process of this algorithm can be illustrated using equation below:

$$E_{K1}(M) = C \quad (2.3)$$

$$D_{K2}(C) = M \quad (2.4)$$

Where K1 and K2 represent the encryption and decryption key respectively.

2.4.3 Symmetric Key versus Asymmetric Key Algorithm

Symmetric key algorithm involves less computation than asymmetric key algorithm and because of that, a symmetric key algorithm is hundreds or thousands of times faster than asymmetric key algorithm. However, there is also a con of using symmetric algorithms. It is less secure compare to asymmetric algorithms because of the "shared secret key".

2.5 Blowfish Algorithm

Since the past few decades, DES was widely used as a standard cryptography algorithm. Unfortunately, the recent design of a $1M machine could recover a DES key in 3.5 hours and this prove that the DES's key size is far too small for today. For such, many approaches has been made, for instant Triple-DES, IDEA, RC4, SAFER and Blowfish. Blowfish algorithm was created by Bruce Schneier to replace the DES algorithm. Blowfish algorithm was first presented at Cambridge Algorithms Workshop with the title "Description of a New Variable-Length Key, 64-bit Block Cipher (Blowfish)" by R. Anderson in 1994. (Bruce Schneier, 1995)

2.5.1 Brief Description of Blowfish Algorithm

Blowfish has a 64-bit block size and a key length of anywhere from 32 bits to 448 bits. In other words, Blowfish is a block cipher that encrypts data in 8-byte blocks.

Blowfish algorithm consists of two parts: a key-expansion part and a data-encryption part. Key expansion converts a variable-length key of maximum 448 bits into several sub-keys arrays totaling 4168 bytes. The data encryption part is a 16-round Fiestel cipher and uses large key-dependent S-boxes. Each round consists of a key-dependent permutation, and a key and data-dependent substitution. This means that in every round, we need to lookup four indexed array data from four s-boxes.

All operations, such as XORs and additions are done on the chunked 32-bit words. The details and methodology of Blowfish algorithm will be discuss in details in next chapter. Figure 2.3 shows the overall operation of Blowfish algorithm.

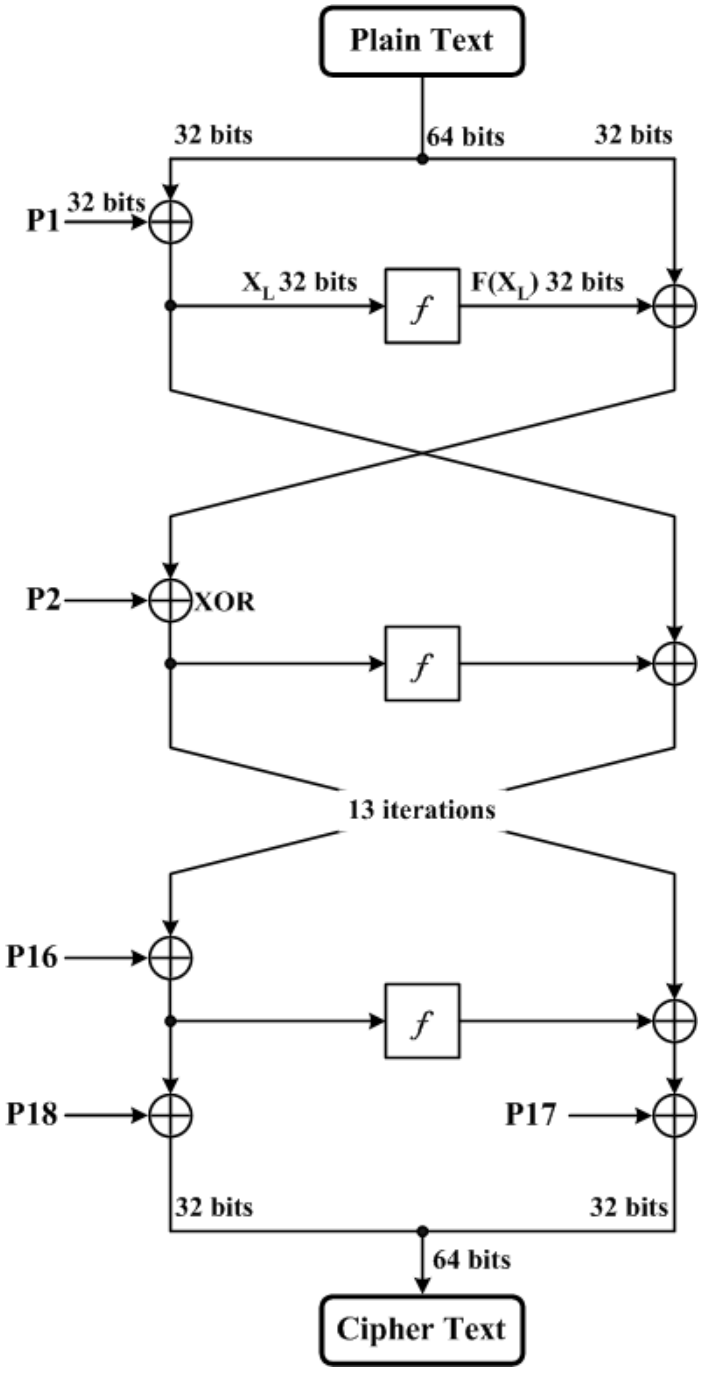

Figure 2.3: The overall operation of Blowfish Algorithm

From figure 2.3, it is clearly shown that Blowfish is a 64 bits block-cipher as the input (plain text) and the output (cipher text) are all 64 bits. The algorithm keeps two types of sub-keys arrays: the P-array and four S-boxes. Every single iteration has one F function. Inside each F function, there are four 256-entry S-boxes. The S-boxes act as a lookup table which will accept 8-bit input and produce 32-bit output. One entry of the P-array is used every round, and after the final round, each half of the data block is XORed with one of the two remaining unused P-entries. Finally, these two results is combined and become the ciphertext.

2.5.2 Advantages of Blowfish Algorithm

Blowfish is a very strong algorithm, because until now it has not been completely cracked. Many designers grow confidence in this algorithm as group after group tries to break it and fails. The nearest attempt is done by Vincent Rijimen who did a second-order differential attack on 4-round Blowfish but cannot be extended to more rounds. (Dr. Dobbs Journal, 1995)

The second advantage of Blowfish is the variable key-length. The key can be varied from 32 bits to 448 bits. So, it is very secure. Although this algorithm is so great, it is easy to implement because it build from simple structure.

The main contributions that make Blowfish algorithm so secure are the non-reversible F-function and the key-dependent S-boxes. The F-function used in this algorithm will give avalanche effect to the Fiestel network. In other words, after every round the key and the plaintext become harder and harder to be cracked. The F function has help Blowfish to become a strong algorithm.

This algorithm uses four S-boxes and it is of course safer than other algorithms which only use one S-box. Furthermore, the S-boxes used in Blowfish is key-dependent. Hence, this algorithm is very safe. To combine the value from all of the S-boxes, Blowfish uses simple addition and XOR.

2.5.3 Sub Section in Blowfish Algorithm

Blowfish algorithm comprises of many important sub sections. They are sub-keys, Fiestel network, s-boxes, p-array, F-function and the encryption part.

- **Sub Keys**

The first and foremost step to use the Blowfish algorithm is generating of sub keys. Sub-keys are divided into p-array and s-boxes.

P-array have 18 sub-keys, all of them are in a chunk of 32 bits.

$P_1, P_2, P_3, P_4, \ldots\ldots, P_{18}$

Whereas each of the four s-boxes has 256 sub-keys. All this sub-keys are in 32 bits.

$S_{1,0}, S_{1,1}, S_{1,2}, \ldots\ldots S_{1,255};$
$S_{2,0}, S_{2,1}, S_{2,2}, \ldots\ldots S_{2,255};$
$S_{3,0}, S_{3,1}, S_{3,2}, \ldots\ldots S_{3,255};$
$S_{4,0}, S_{4,1}, S_{4,2}, \ldots\ldots S_{4,255};$

- **Fiestel Network**

In encryption, many modern symmetric block ciphers are based on Feistel networks, including Blowfish. It is also commonly known as a Feistel cipher. A Fiestel cipher, which is named after IBM cryptographer Horst Feistel, is a block cipher with a particular structure. For Feistel structure, the encryption and decryption operations are very similar, even identical in some cases, requiring only a reversal of the key schedule. Therefore the size of the code or circuitry required to implement such a cipher is nearly halved. In fact, this is the primary advantage of the Fiestel network. The iterative nature of the Feistel construction makes implementing the cryptosystem in hardware easier because it use repeated operations for each iteration such as bit-shuffling or often called

permutation box, simple non-linear functions or often called Substitution boxes and linear mixing which used XOR.

In Fiestel network, the input text or plaintext will be divided by two to become L and R. So, the output from the *i*-th round depends on the output from the previous round. The equation of Fiestel network is shown below.

$$L_i = R_{i-1} \tag{2.5}$$

$$R_i = L_{i-1} \text{ XOR } f\,(R_{i-1}, K_i) \tag{2.6}$$

Where K_i represents the key used in *i*-th round and *f* is the arbitrary round function or F-function.

This function is reversible and hence, the encryption and decryption process can be done using the same algorithm and same key. Figure 2.4 shows the operations of a Fiestel Cipher.

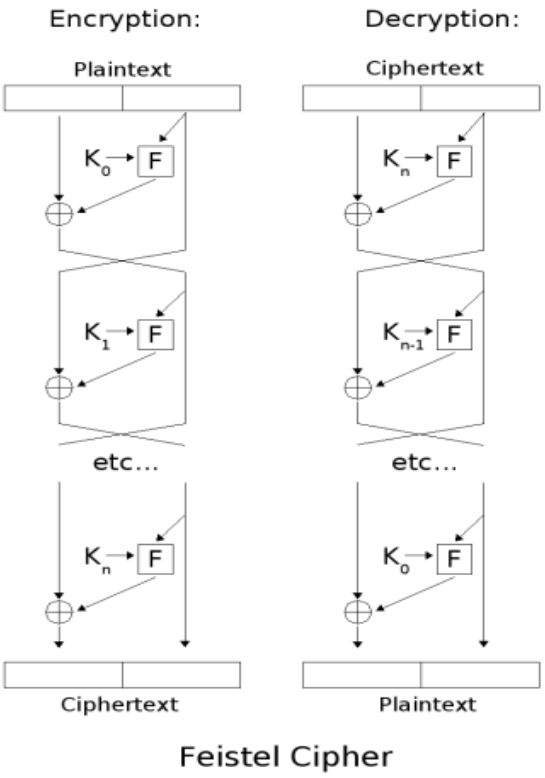

Figure 2.4: Fiestel Cipher

- **Arbitrary Round Function or F-function**

F function appears in every iteration of the Blowfish operation. This function also known as arbitrary round functions. In encryption, this F function is very important and it is nonreversible. The function splits the 32 bits input into four eight-bit quarters, and uses the quarters as input to the S-boxes. The S-boxes will determine and choose a 32-bits output. The outputs from the four S-boxes are then added modulo 2^{32} and XORed to produce 32 bits output. Figure 2.5 shows the operation of the F function.

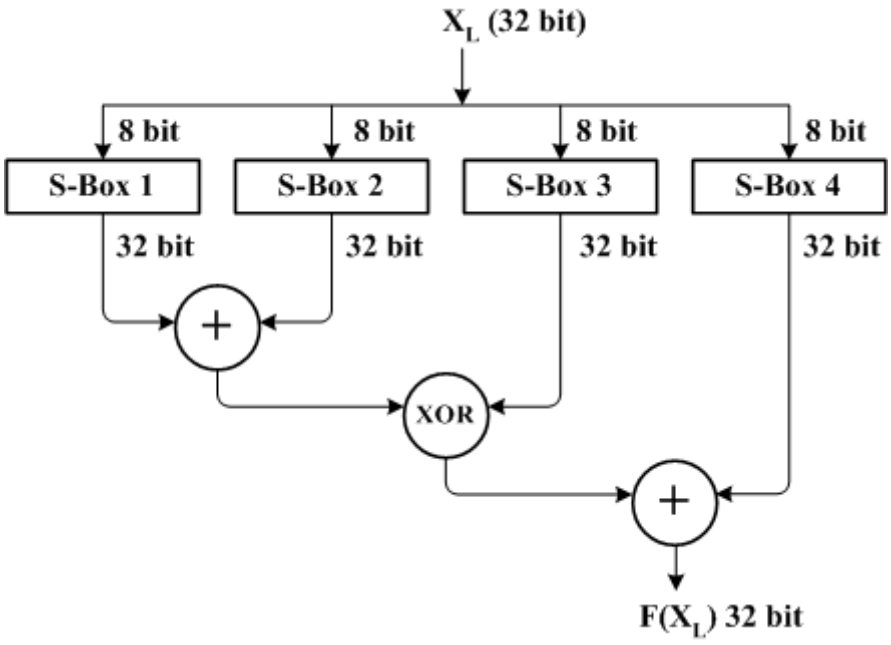

Figure 2.5: Blowfish's F function

2.5.4 Products that used Blowfish Algorithm

Blowfish has proven to be one of the safest and not patented encryption algorithms. Therefore, there are plenty of products that used Blowfish encryption algorithm to enhance its security. Some of the popular products are listed in the table 2.1 below.

Table 2.1: Products that used Blowfish Algorithm

Product	Producer	Function
96 Crypt	Fever.link	A file and folder encryption/decryption program.
A-Lock	Trillium Technology Group	Encryption software that integrates with popular Windows e-mail programs.
Cryptlib	Peter gutmann	A security toolkit that allows programmers to add encryption and authentication services to their software.
HTTPort	Technology Networks LLC	TCP/IP through HTTP tunneling software that allows users to bypass virtually any HTTP proxy and tunnel arbitrary TCP/IP through it. From version 3 it uses Blowfish to encrypt traffic.
Linux	No information	Includes Blowfish in the mainline kernel, starting with v2.5.47
ISequre	BD Universal Software	A Windows program that encrypts ICQ messages.
Password safe	Counterpane System	A free password database with strong encryption.
MetaPass Digital Key	MetaPass, Inc.	A USB key that stores passwords for automatic login to websites, applications, and terminal-based devices.
PGPFone	ETEL	A secure Internet phone application. Uses triple-DES or Blowfish to encrypt the voice stream.
Password Plus	DataViz	Password management for Windows, Macintosh, handheld devices, and smartphones.
Safe Guard Easy	Utimaco	Disk encryption for DOS, OS/2, and Windows 3.x, 9x, and NT.

2.6 Summary

In short, this chapter touched the theory part and the basic knowledge that is acquired in order to proceed further in this project. For instant, the encryption algorithm, Blowfish algorithm, VHDL and FPGA.

CHAPTER 3
METHODOLOGY

3.1 Introduction

Steps to implement Blowfish Algorithm will be presented in this chapter. On the beginning part of this chapter, the mathematical equations and the theory part of Blowfish is mentioned in detail with the help of flowcharts. After that, the software and the hardware used will be explained in detail, ranging from the steps to produce the design entry until connecting the hardware to the computer in order to download the program to the FPGA board. The last part of this chapter, which is also the core part, will explain the VHDL coding part by part in details.

3.2 Key Expansion

The initial values for P-array and sub-keys are fixed by Bruce Schneier, the founder of Blowfish. However, to further generate the sub-keys, the original key need to go through all steps in Blowfish algorithm.

3.2.1 Generation of the sub-keys

For Blowfish algorithm, the key can be any length from 32 to 448 bits, in steps of 32 bits. This means that a 64-bit key is accepted while a 70-bit key is not accepted. The first method in the algorithm is to break the original key into a set of sub-keys. Blowfish uses a large number of sub-keys. These keys must be pre-computed before any encryption or decryption process can be performed.

As mentioned in chapter 2, the P-array contains 18 32-bit sub-keys while each of the four S-boxes contains 256 32-bit sub-keys.

First, initialize the P-array and then the four S-boxes with a fixed string. This must be done in order. After initialize the P-array, then only can initialize S-box1. S-box2, S-box3 and S-box4 are initialized using this method too. The string consists of the hexadecimal digits of π but must less the initial digit 3. (Schneier, 1994) For example:

P_1 = 243f6a88

P_2 = 85a308d3

P_3 = 13198a2e

P_4 = O3707344

. . .

$S_{[4,254]}$ = 578FDFE3

$S_{[4,255]}$ = 3AC372E6

After the value of the P-array is initialized, the first 32 bits of the key is XOR with P_1, the second 32-bits of the key is XOR with the P_2 and this continues for all bits of the key. Repeatedly cycle through the key bits until the entire P-array has been XORed with key bits.

For example, if the key size is 448-bits, the key is chunked into 14 32-bit division, namely K_1, K_2, … , K_{14}. This chunked-key is then XOR with all the P-array.

$P_1 = P_1 \text{ XOR } K_1$

$P_2 = P_2 \text{ XOR } K_2$

. . .

$P_{14} = P_{14} \text{ XOR } K_{14}$

$P_{15} = P_{15} \text{ XOR } K_1$

. . .

$P_{18} = P_{18} \text{ XOR } K_4$

For another example, if the key size used is 128-bits, it is divided into 4 32-bits words K_1, K_2, K_3, K_4 and the following is what happens:

$P_1 = P_1$ XOR K_1

$P_2 = P_2$ XOR K_2

$P_3 = P_3$ XOR K_3

$P_4 = P_4$ XOR K_4

$P_5 = P_5$ XOR K_1

$P_6 = P_6$ XOR K_2

. .

$P_{18} = P_{18}$ XOR K_2

The following step is to encrypt the all-zero string with the Blowfish algorithm, using the new sub-keys, P_1 until P_{18}, described just now, and call it output (a).

The sub-keys are modified by replacing P_1 and P_2 with the 64-bit output (a). This means that the first 32-bit replace the value of P_1 and the second 32-bit replace the value of P_2.

Now, encrypt again using the same Blowfish algorithm with the modified sub-keys, and will get another set of output, call it output (b). Then, replace P_3 and P_4 with the 64-bit output (b).

Continue the process for 521 times, replacing all entries of the P-array, and then all four S-boxes in order, with the output of the continuously-changing Blowfish algorithm.