AN EMBEDDED SYSTEM FOR NETWORKING SECURITY APPLYING CRYPTOGRAPHIC ACCELERATION IN FIELD PROGRAMMABLE GATE ARRAY HARDWARE

VISHNU A/L PARAMASIVAM

UNIVERSITI TEKNOLOGI MALAYSIA

AN EMBEDDED SYSTEM FOR NETWORKING SECURITY APPLYING CRYPTOGRAPHIC ACCELERATION IN FIELD PROGRAMMABLE GATE ARRAY HARDWARE

VISHNU A/L PARAMASIVAM

A thesis submitted in fulfilment of the
requirements for the award of the degree of
Master of Engineering (Electrical)

Faculty of Electrical Engineering
Universiti Teknologi Malaysia

NOVEMBER 2009

Dedicated to
my beloved family

ACKNOWLEDGEMENT

First and foremost, I would like to extend my deepest gratitude to my project supervisor, Prof. Dr. Mohamed Khalil bin Mohd Hani, for giving me the opportunity to work the area of work that interests me the most. His constant encouragement, critics and guidance were key to bringing this project to a fruitful completion. I have learned and gained much, not only in the field of research, but also in the lessons of life.

My sincerest appreciation goes to my co-supervisors, Prof. Madya Munim Zabidi and Dr. Nadzir Marsono, and my seniors Jasmine Hau Yuan Wen and Ilyasak for their support, help, and technical advices. I have learned much from them, as well as receiving plenty of guidance and motivation.

I would also like to thank all those who have contributed directly and indirectly to the completion of this research and thesis. This includes my fellow postgraduate students, Mohd Nazrin and Arif Irwansyah, who provided me with help and company during my study here.

Finally, I would like to thank my family for always being there for me, through thick and thin. Their role in my life is something I will always need.

ABSTRACT

The Internet is an insecure medium. The Secure Socket Layer (SSL) protocol and its successor Transport Layer Security (TLS) can be used to secure applications that communicate over a network. The most widely deployed, freely available implementation of the SSL/TLS protocol is the OpenSSL library. When using the SSL/TLS protocol, the computational power required is typically too much for most embedded systems, because cryptographic functions are computationally extensive. The solution to this problem would be to perform hardware acceleration of computationally intensive cryptographic functions. This thesis proposes an embedded cryptosystem with Field Programmable Gate Array based hardware acceleration for networking security, applying the OpenSSL cryptographic protocol. The key cryptographic functions used in SSL/TLS-driven connections are Advanced Encryption Standard (AES), Secure Hash Algorithm (SHA), Rivest-Shamir-Adleman (RSA), and Random Number Generation (RNG). The AES hardware symmetric cryptographic hardware core is newly designed, the SHA-1, SHA-2, RNG, and RSA cores are improved from previous work, and the system bus interface of these hardware cores are upgraded. All of these hardware cores are integrated into an embedded system implemented as a System-on-Chip. Finally, the OpenSSL cryptographic library is accelerated using this cryptosystem to improve the performance of networking security. Nios2-Linux Real Time Operating System is used within the embedded system. It provides native support for Ethernet, Universal Serial Bus, multitasking, standard Linux functions, and has a large collections of ready-to-use libraries, which includes the OpenSSL library. Applications are written to test, verify, and benchmark the embedded cryptosystem. Results show an improvement in performance by 9 to 278 times of the OpenSSL crypto library, depending on the algorithm accelerated. The performance for networking security using the SSL/TLS protocol through the OpenSSL library is also improved.

ABSTRAK

Internet adalah suatu bahantara yang tidak selamat. Protokol *Secure Socket Layer* (SSL) dan penggantinya *Transport Layer Security* (TLS) biasanya digunakan untuk aplikasi yang perlu berkomunikasi secara selamat dalam rangkaian Internet. Sistem pelaksanaan protokol SSL/TLS yang digunakan secara meluas dan percuma adalah melalui rutin perpustakaan OpenSSL. Bila menggunakan protokol SSL/TLS, kuasa pengiraan yang dikehendaki lazimnya terlalu tinggi untuk sistem komputer terbenam disebabkan fungsi-fungsi kriptografi yang kompleks. Penyelesaian yang sesuai untuk masalah ini adalah dengan menggunakan pecutan litar logik untuk fungsi-fungsi kriptografi yang intensif. Tesis ini mencadangkan satu sistem kripto terbenam dengan mengunakan *Field Programmable Gate Array* yang berasaskan pecutan litar logik untuk keselamatan perangkaian perhubungan dengan mengaplikasikan protokol kriptografi OpenSSL. Fungsi-fungsi kriptografi yang penting dalam sambungan SSL/TLS adalah *Advanced Encryption Standard* (AES), *Secure Hash Algorithm* (SHA), *Rivest-Shamir-Adleman* (RSA), dan *Random Number Generation* (RNG). Pecutan perkakasan untuk fungsi kriptografi AES direka, pecutan litar logik SHA-1, SHA-2, RNG, dan RSA dinaik taraf daripada projek-projek sebelumnya, dan sistem antaramuka untuk litar-litar logik tersebut juga dinaik taraf. Kesemua litar logik ini disepadukan dalam sebuah sistem terbenam yang dikenali sebagai *System-on-Chip*. Akhirnya, rutin perpustakaan kriptografi OpenSSL dipecut dengan menggunakan sistem kripto ini untuk meningkatkan kelajuan keselamatan perangkaian. Sistem pengendalian masa sebenar Nios2-Linux digunakan dalam sistem terbenam ini. Ia menyediakan infrastruktur untuk Ethernet, *Universal Serial Bus*, pengendalian berbilang tugas, fungsi-fungsi lazim Linux, dan koleksi besar rutin perpustakaan yang sedia digunakan, termasuk OpenSSL. Beberapa aplikasi ditulis untuk menguji, mengesah, dan mengukur kelajuan sistem kripto terbenam ini. Berdasarkan keputusan yang diberi oleh applikasi ini, peningkatan sebanyak 9 hingga 278 kali ganda diperhatikan dalam pengedalian rutin perpustakaan OpenSSL. Secara tidak langsung, kelajuan keselamatan perangkaian SSL/TLS melalui rutin perpustakaan OpenSSL juga dipertingkatkan.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

3DES	–	Triple Data Encryption Algorithm
AES	–	Advanced Encryption Standard
API	–	Application Programming Interface
ASIC	–	Application Specific Integrated Circuit
BN	–	Big Number
CA	–	Certificate Authority
CPU	–	Central Processing Unit
DES	–	Data Encryption Standard
ECC	–	Elliptic Curve Cryptography
FIFO	–	First In First Out
FIPS	–	Federal Information Processing Standard
FPGA	–	Field Programmable Gate Array
FSM	–	Finite State Machine
GPU	–	Graphic Processing Unit
GUI	–	Graphical User Interface
HAL	–	Hardware Abstraction Layer
HDL	–	Hardware Development Language
IC	–	Integrated Circuit
I/O	–	Input/Output
IP	–	Intellectual Property
LE	–	Logic Elements
Mbit	–	Mega Bits
MD5	–	Message Digest Algorithm (Fifth Series)

MHz	–	Mega Hertz
ms	–	millisecond
PC	–	Personal Computer
PCI	–	Peripheral Component Interconnect
PIO	–	Parellel Input Output
PKI	–	Public Key Infrastructure
RISC	–	Reduced Instruction Set Computer
RNG	–	Random Number Generator
ROM	–	Read Only Memory
RSA	–	Rivest-Shamir-Adleman
RTL	–	Register Transfer Level
SDRAM	–	Synchronous Dynamic Random Access Memory
SHA-1	–	Secure Hash Algorithm - 1
SHA-2	–	Secure Hash Algorithm - 2
SoC	–	System-on-Chip
SOPC	–	System-on-Programmable-Chip
SSL	–	Secure Sockets Layer
TLS	–	Transport Layer Security
UART	–	Universal Asynchronous Receiver Transmitter
UNIX	–	Uniplexed Information and Computing System (originally spelled UNICS)
USB	–	Universal Serial Bus
VHDL	–	Very High Speed Integrated Circuit Hardware Description Language
VLSI	–	Very Large Scale Integration
VoIP	–	Voice over Internet Protocol

LIST OF APPENDICES

CHAPTER 1

INTRODUCTION

This chapter provides background information, the problem statement, objectives, work scope, methodology, and research contribution of this thesis, which is about building an embedded cryptosystem for network security applying hardware acceleration in Field Programmable Gate Array (FPGA).

1.1 Background

The Internet is an insecure medium [1]. It is possible to intercept and modify data on the Ethernet wire, and the process of doing so is not complicated. An average hacker will have an easy time eavesdropping if the data transferred is sent in plain-text or other standard formats. This can be done using *man in the middle attacks*, as depicted in Figure 1.1. This is because data on a network is broadcast, hence even amateur hackers can listen in. Hence, applications that do not properly protect data when using an untrusted medium are very vulnerable.

Network security consists of the provisions made in an underlying computer network infrastructure, policies adopted by the network administrator to protect the network and the network-accessible resources from unauthorized access and the effectiveness (or lack) of these measures combined together. Currently, the most widely deployed method for Internet security is the Secure Sockets Layer (SSL) cryptographic protocol. The most popular implementations are built using the OpenSSL library.

SSL is an excellent protocol, but it is a lot slower than a traditional unsecured TCP/IP connection [1]. Cryptographic functions on embedded systems

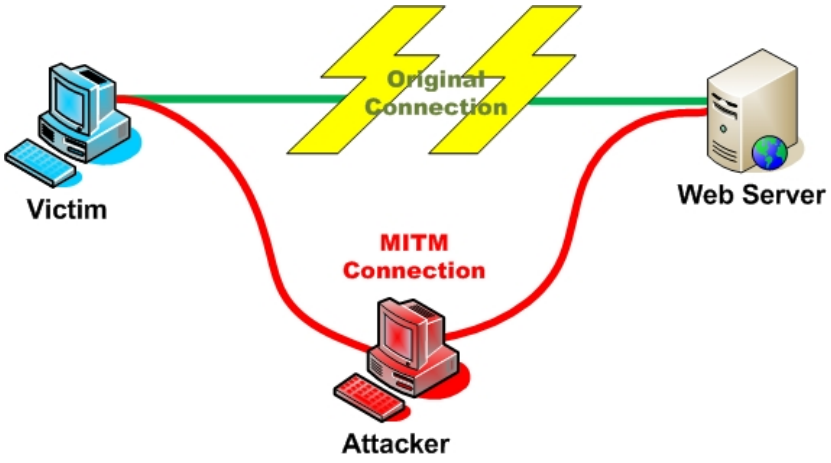

Figure 1.1: Illustration of a man in the middle attack.

are computationally extensive, especially asymmetric cryptography such as Rivest-Shamir-Adleman (RSA) for key generation. Symmetric cryptography like Advanced Encryption Standard (AES) and Triple Data Encryption Algorithm (3DES) works on a per block basis. Every data transferred must be encrypted/decrypted. Therefore, when using the SSL cryptographic protocol, the computational power required is typically too much for most embedded systems [2]. This includes the Altera Nios II which only runs on a 50MHz clock.

This is important, because the CPU has to spend its resources on doing other tasks. One particularly good example would be Voice over Internet Protocol (VoIP) and credit card transactions. Voice data has to be transferred as fast to deliver seamlessly, the data must be encrypted in real time. This is not possible without cryptographic acceleration. On a different take, credit card transactions are very slow due to the use of multiple cryptographic protocols (especially RSA or ECC), typically taking 20-40 seconds for a 50MHz embedded system [2]. A cash transaction would definitely be faster. An embedded system with cryptographic acceleration would definitely be useful in these situations, because a single transaction would take less than a second, assuming there is no network congestion.

Another advantage of an embedded system is that hardware computations are definitely much safer than that done in software. This is due to attacks that use timing analysis, known as the timing attack [3]. In cryptography, a timing attack is a side channel attack in which the attacker attempts to compromise a cryptosystem by analyzing the time taken to execute cryptographic algorithms. Every logical operation in a computer takes time to execute, and the time can differ based on the input; with precise measurements of the time for each operation, an attacker can work backwards

to the input. Information can leak from a system through measurement of the time it takes to respond to certain queries. How much such information can help an attacker depends on many variables: cryptosystem design, the CPU running the system, the algorithms used, assorted implementation details, timing attack countermeasures, the accuracy of the timing measurements, etc.

Timing attacks are often overlooked in the design phase because it is very dependent on the implementation [3]. Software implementations use several methods to prevent this type of attack. For example, RSA uses a method known as blinding. RSA blinding uses extra processing time, but effectively 'blinds' a timing attack by changing the amount of time taken for the RSA operation. However, with accelerated hardware cores, the amount of time taken for by hardware accelerated cores are the same regardless of the data processed. In other words, the amount of clock cycles taken for any cryptographic operation done in hardware is set in stone. So technically, a timing attack is useless to make an accurate prediction.

1.2 Problem Statement

Firstly, data security comes at a high cost for embedded systems. The performance of most embedded CPUs (such as ARM and Nios II) are just not adequate enough to perform cryptography as well as do other tasks such as image or signal processing.

Secondly, the most computer intensive aspect of networking security is cryptography. This becomes very apparent when high strength cryptography is applied in the security protocol. High strength cryptography is typically required in banking and military based applications.

A possible solution to this problem would be to perform hardware acceleration of the computer intensive cryptographic functions. The advantage of this approach is, not only performance is enhanced, but also the level of security is increased, since the cryptographic functions are computed in hardware. In computing, hardware acceleration is the use of hardware to perform some function faster than is possible in software running on a general purpose CPU. Examples of hardware acceleration include blitting acceleration functionality in graphics processing units (GPUs) and instructions for complex operations in CPUs.

A good cryptographic protocol is vital for ensuring security to provide confidentiality, authenticity, integrity, and non-repudiation. The Secure Socket Layer (SSL) protocol and its successor Transport Layer Security (TLS) can be used to secure web based applications that need to communicate over a network [1].

The key cryptographic functions used in SSL-driven connections are AES, SHA-1, SHA-2, RSA, and random number generation. Theoretically, if these functions are accelerated, it will greatly increase the speed of an SSL connection. All these functions can be accelerated, by designing hardware cryptographic accelerator cores. These hardware cores are designed using HDL and can be deployed in Field Programmable Gate Array (FPGA) based systems.

To work with an embedded system supported with the OpenSSL library, Ethernet, and Universal Serial Bus (USB) a real time operating system is needed. There are many freely available embedded operating systems available in the market, but Nios2-Linux (based on μClinux) is the most compatible one to be applied in this work. Nios2-Linux provides native support for Ethernet, TCP/IP stack, USB, multitasking, standard UNIX functions, and has a large collections of ready-to-use libraries, which includes the OpenSSL library.

1.3 Objectives

1. To design an embedded cryptosystem with FPGA-based hardware acceleration for networking security, applying the OpenSSL cryptographic protocol. This involves the following sub-objectives:
   (a) Customize the design of the AES hardware symmetric cryptographic hardware core to be dynamic and able to cater for every type of situation.
   (b) Upgrade and modify the SHA-1, SHA-2, RNG, and RSA cores so that they are faster and better suited to work with the OpenSSL library.
   (c) Improve the system bus interface of these hardware cores.
   (d) Integrate all the above cores into an embedded system implemented as a System-on-Chip (SoC), as depicted in Figure 1.2.
   (e) Accelerate the OpenSSL cryptographic library using this cryptosystem to improve the performance of networking security.

2. To design the following applications using the embedded cryptosystem proposed in the first objective:

   (a) A complete cryptographic function test program.

   (b) Secure bank check transfer system using networking security.

   (c) File encryption and decryption program using the AES algorithm.

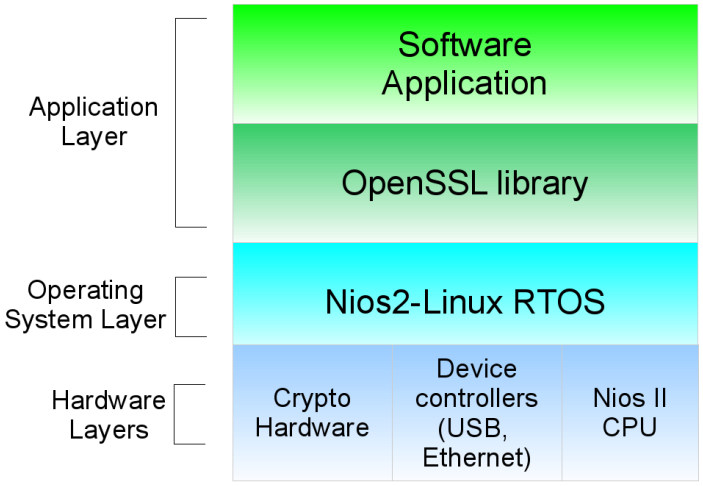

Figure 1.2: Layers of the proposed embedded cryptosystem.

1.4 Scope of Work

- Verilog HDL is applied in the design of cryptographic hardware cores and the interface to the Nios II processor.
- The OpenSSL library is used for key exchange, symmetric block cipher, asymmetric block cipher, and hashing algorithms.
- Nios2-Linux operating system is deployed in the embedded system to complement the OpenSSL library with Ethernet, USB, and filesystem functionality.
- Test applications are limited to C/C++ programs linked to the OpenSSL library.
- Cryptographic acceleration is limited to the following algorithms:
  - RSA 512/1024/2048 bits.
  - SHA 160 bits.
  - SHA 224/256/38/512 bits

- AES 128/192/256 bits
- Random Number Generation.

- The prototype is designed and implemented on the Altera Stratix II FPGA development board, with the Nios II embedded processor running at 50 MHz.

1.5 Methodology

This project integrates cryptographic (crypto) acceleration into embedded systems for use in networking security. The embedded system environment proposed in this work consists of a FPGA-based system on chip running on the Nios II CPU, coupled with hardware crypto accelerator cores for AES, SHA-1, SHA-2, RSA, and a random number generator (RNG). The hardware runs on Nios2-Linux (real-time operating system), which is configured with Ethernet connectivity, USB functionality, and a working build of the OpenSSL library. The hardware cores are directly linked to the OpenSSL cryptographic functions, and effectively accelerate all incoming and outgoing SSL secured connections.

Figure 1.3 illustrates the FPGA-based embedded system connected to a PC through an untrusted Ethernet medium. It uses SSL connections for data communication with the PC. The OpenSSL library is deployed in this project because it is open-source, free, and works on any Linux based operating system.

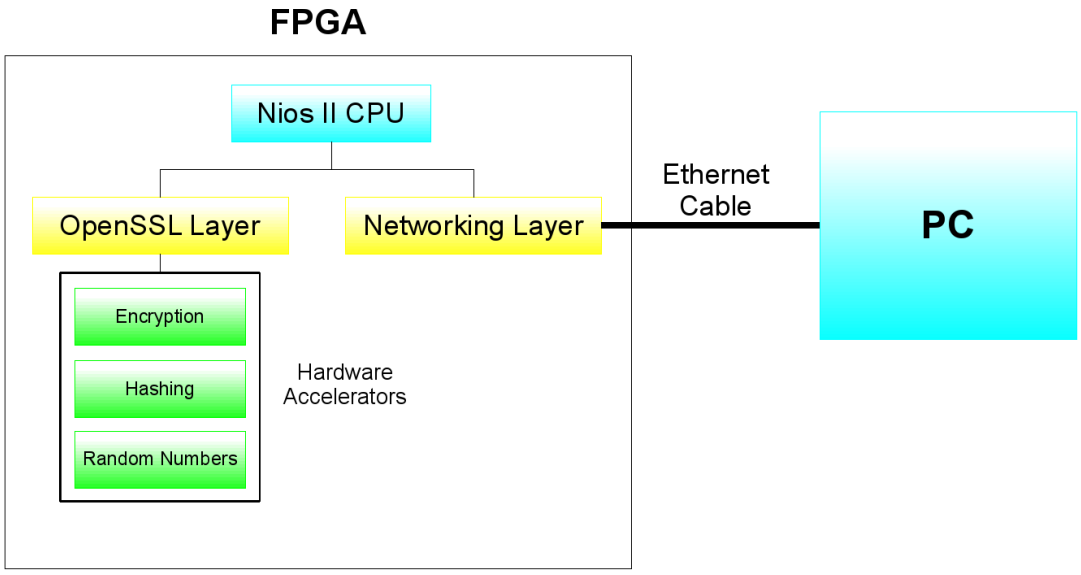

Figure 1.3: Secure communication between the embedded system and a PC.

Figure 1.4 is a diagram depicting the approach to design the embedded cryptosystem. To summarize the illustration, the approach taken comes down to preparing all the cryptographic cores and integrating them into an SoC. At the same time the Nios2-Linux operating system and the OpenSSL library is prepared. Finally the system is setup and a security application for it is designed. Finally, the speed improvement is measured and the results are reported.

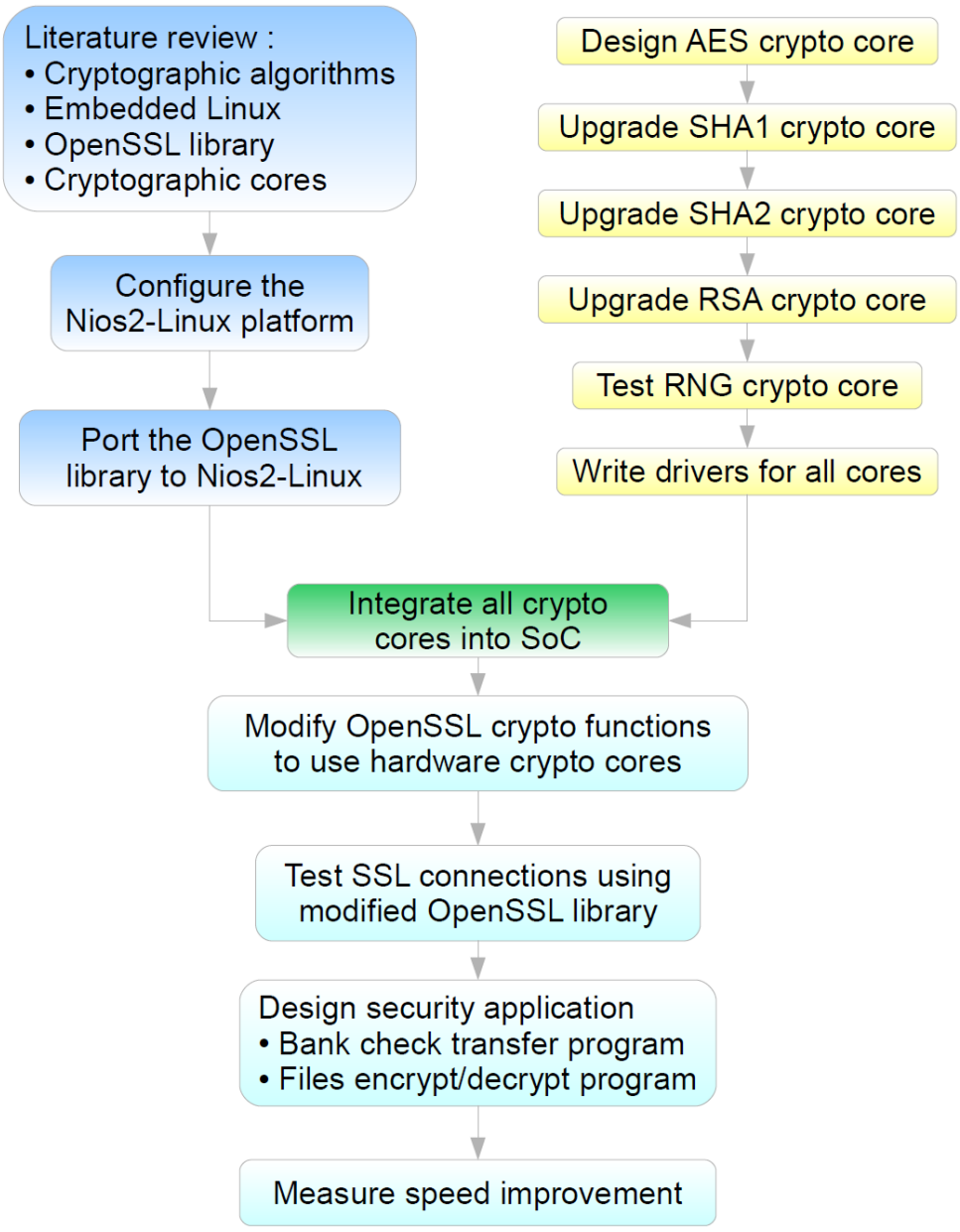

Figure 1.4: Methodology used to achieve research objectives.

1.6 Research Contribution

1. The major contribution of this research is that it is the only work done to accelerate the OpenSSL cryptographic library in embedded systems. Therefore the speed improvements can only be measured against itself (running software routines) and other similar works such as [4].
2. Laying the basic foundation for secure embedded systems through Nios2-Linux and the OpenSSL library. This provides a TCP/IP stack, networking security, USB host functionality, filesystem stack, as well as open up the possibilities of porting and using Linux based open source libraries.
3. Improving available cryptographic hardware cores, as well as pioneering the integration of hardware coprocessors and the OpenSSL software library. This provides accelerated networking security.
4. Detailing methods of how to optimize data transfer from CPU to hardware coprocessor. Methods used are endian switching, designing a robust Avalon interface, and eliminating unnecessary communication.

1.7 Thesis Organization

This thesis is organized into seven chapters. The first chapter is the introductory chapter. It presents background information, the problem statement, objectives, work scope, methodology, and research contribution of this thesis.

Chapter 2 is the literature review and background chapter. It presents the basic theory of the building blocks used in this project such as embedded Linux, cryptographic algorithms, and communication mediums such as Ethernet and USB.

Chapter 3 gives an overview of the embedded cryptosystem proposed in this thesis. This includes an introduction to the system as well as the applications targeted for it, the layers of the system based on the TCP/IP model, and the Nios2-Linux RTOS implementation in the Altera FPGA-based development system.

Chapter 4 is the hardware design chapter. It explains the design of the components in the hardware layer of the proposed embedded cryptosystem. This

includes all the cryptographic hardware cores, the bus interface, and all the hardware modules connected to it.

Chapter 5 is the OpenSSL and software subsystem chapter. It describes the integration the OpenSSL library in the proposed embedded cryptosystem. This includes the porting of the OpenSSL library to the Nios2-Linux platform, the modifications performed on the cryptographic library source code, certificate generation, and the integration of the library with the hardware cryptographic cores.

Chapter 6 is the results chapter. It describes the verification and the speed improvement of all the hardware cryptographic cores. Implementation of several real applications that use cryptography are also described here.

The final chapter is the conclusion chapter, which concludes the findings of this project and discusses the potential future work to further enhance the overall embedded system.