8192-BIT RIVEST-SHAMIR-ADLEMAN DATA ENCRYPTION HARDWARE ACCELERATOR

CHEW YEN WEN

A project report submitted in partial fulfilment of the
requirements for the award of the degree of
Master of Engineering (Electrical – Computer & Microelectronic System)

Faculty of Electrical Engineering
Universiti Teknologi Malaysia

DECEMBER 2015

This is dedicated to my beloved family.

ACKNOWLEDGEMENT

I would like to express my gratitude to my current supervisor, Assoc. Prof. Dr. Muhammad Nadzir Bin Marsono for guiding me towards the completion of this project. He generously spending time to supervise me and to give timely advices. I would also like to thank my previous supervisor, Dr. Rabia Bakhteri for her guidance and inspiration of the project topic. I express my deep sense of gratitude to my manager, Lai Soon Chong for being supportive throughout my part-time Master study. He was being helpful by providing technical advices when I have problems. Sincere thanks to my company, Intel for sponsoring my part-time Master study at Universiti Teknologi Malaysia (UTM). I am grateful towards my classmates, colleagues and friends who have directly or indirectly helped me along the duration of my Master study. Lastly, this project would not have been possible without the moral support and encouragement from my family. I would like to thank my family for being there for me.

ABSTRACT

Rivest-Shamir-Adelman (RSA) algorithm is one of the state-of-art public-key cryptography that is efficient in terms of implementation because it uses the same general equation for encryption and decryption, that is, modular exponentiation equation. The security reliability of RSA algorithm is based on the difficulty of factoring a large number. The larger the RSA key size, the higher the security level that can be achieved. However, at the same time, the complexity of the computation increases, which results in more computation cycles. Software implementation of RSA with large key size is too slow and less effective for large amount of data encryption or decryption. Hence, the purpose of this project is to implement a hardware-based RSA coprocessor to handle RSA encryption and decryption effectively. This project implements a RSA coprocessor using radix-2 Montgomery modular multiplication that described at bit-level. This implementation uses carry-saved adders to achieve parallel processing in hardware. The hardware implementation of the RSA coprocessor is done using Verilog synthesizable Register-transfer Level (RTL) code to allow scalability. Simulation results are obtained to validate the functionality of the design. The design is synthesized using Altera Quartus software tool to evaluate the performance of the implementation. The designs are synthesized on device Stratix V 5SEEBF45I4 for key-size of 128-bit, 1024-bit and 8192-bit. The data throughput of the 8192-bit design can reach up to 3.387 kbps with LE utilization of 30% on the device used. Although the performance of the design is not the highest among the related works, but this design provides a proven working prototype for 8192-bit RSA coprocessor using Bit-level Montgomery Modular Multiplication for hardware parallel processing.

ABSTRAK

Rivest-Shamir-Adelman (RSA) algoritma adalah salah satu kripto algoritma yang menggunakan kekunci umum. RSA cekap dari segi pelaksanaan kerana ia menggunakan persamaan umum yang sama untuk penyulitan dan penyahsulitan, iaitu, persamaan pengeksponenan modular. Keselamatan algoritma RSA adalah berdasarkan kesukaran memperfaktorkan nombor yang besar. Semakin besar saiz kekunci RSA, semakin tinggi tahap keselamatannya. Walau bagaimanapun, pada masa yang sama, kerumitan pengiraan juga meningkat dan menyebabkan lebih banyak kitaran pengiraan. Pelaksanaan RSA dengan saiz kunci yang besar menggunakan perisian adalah perlahan dan kurang berkesan untuk data yang besar. Oleh itu, tujuan projek ini adalah untuk menghasilkan kopemproses RSA berasaskan perkakasan supaya dapat mengendalikan penyulitan dan penyahsulitan RSA dengan lebih cekap. Projek ini menghasilkan kopemproses RSA menggunakan pendaraban modular Montgomery radiks-2 dalam tahap bit. Perlaksanaan projek ini menggunakan penambah simpan-bawa untuk mencapai pemprosesan selari dalam perkakasan. Kod tahap daftar data (RTL), Verilog digunakan untuk menghasilkan rekaan perkakasan ini supaya hasil reka ini lebih berskala. Hasil penyelakuan diperolehi untuk mengesahkan kefungsian rekaan projek ini. Rekaan projek ini disintesis dengan menggunakan Altera Quratus untuk menilai prestasinya.Hasil reka ini disintesis dengan menggunakan peranti Stratix V 5SEEBF45I4 untuk saiz kekunci 128 bit, 1024 bit dan 8192 bit. Daya pemprosesan hasil reka bit 8192 boleh mencapai 3.387kbps dengan penggunaan logik 30% atas peranti tersebut. Walaupun prestasi hasil kerja ini bukan yang tertinggi antara kerja-kerja sebelum yang berkaitan, namum hasil kerja ini menyediakan prototaip kopemproses RSA 8192-bit berfungsi yang menggunakan pendaraban modular Montgomery radiks-2 dalam tahap bit untuk pemprosesan selari dalam perkakasan.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

ALM	-	Adaptive Logic Modules
ASM	-	Algorithm State Machine
ECC	-	Elliptic Curve Cryptography
FBD	-	Functional Block Diagram
GCD	-	Greatest Common Divisor
L-R Binary	-	Left-to-Right Binary
PE	-	Processing Element
R-L Binary	-	Right-to-Left Binary
RSA	-	Rivest-Shamir-Adelman
RTL	-	Register-transfer Level
Verilog HDL	-	Verilog Hardware Description Language

LIST OF APPENDICES

CHAPTER 1

INTRODUCTION

1.1 Problem Background

Since the invention of computer and internet connections, the way of people store and communicate information has changed drastically from physical forms to digital forms. From one of the latest report of EMC-sponsored IDC Digital Universe study (Gantz and Reinsel, 2012), an estimation of 2.8 zettabytes of data is created and replicated in year 2012. The study also projected that by 2020, the amount of data in the digital world will reach 40 zettabytes. With such a huge amount of information accessible over the wire, the issue of privacy and data security become a major concern. It is estimated that about one third of the data in the digital world requires a certain extend of security for the purpose of privacy, regulations and fraud prevention. The examples of data that required high security are banking information, corporate information, personal account information, and payment transaction.

In recent years, facts like the wide acceptance of online shopping activities, the significant growth in smart mobile devices, and the fast-paced software development have made security a basic requirement in global computing ecosystem. On top of the data security concerns in client and server computing systems, the growing wave of Internet of Things (IoT) recently has again surfaced the demands on internet security. The IoT extends existing internet infrastructure to embedded computing devices that realized machine-to-machine communications, environmental monitoring and control, smart applications, and telehealth applications. Without security stacks in the application, hackers can easily alter the data and take over control of the application.

To protect information from unauthorized parties, data security system is implemented. The backbone of a data security system is data cryptography. Up to date, there are numerous of data cryptography algorithms available to serve the similar purpose, that is, to protect information by encrypting it. Rivest-Shamir-Adleman (RSA) algorithm is one of the cryptography algorithm that is widely used. The RSA algorithm has proven to be highly secured although the algorithm is relatively more complex than symmetric-key cryptography algorithm. The security level of RSA algorithm can be increased by using a larger key size. However, as the key size increases, the computation complexity of the data cryptography increases as well. This will cost the speed of the data cryptography process.

For some of the applications where the speed of data cryptography and the level of security are equally important, hardware-assisted cryptography system is a good solution. The complex computation part of the RSA cryptography can be leveraged to a dedicated hardware RSA coprocessor. With this approach, the speed of data cryptography can be kept at an acceptable level even for large key-size. However, implementation of RSA algorithm in hardware is not straight forward. The algorithm need to be modified in order to be implemented in hardware. There are several methods that can be used to implement RSA algorithm, with each of it has different advantages in terms of performance and resources.

1.2 Problem Statement

RSA public key cryptography algorithm has been widely implemented for data security solution due to its high level of reliability and security. The uniqueness of RSA algorithm is that both the encryption and decryption processes used the same mathematical operation. However, the biggest drawback of the algorithm is the long computation time due to its underlying complex wide-operand modular arithmetic. The larger is the RSA keys' size, the higher level of security it can achieve. However, at the same time, the complexity of the algorithm increases, which results in more computation cycles. As the computation system power growing speedily from year to year, a relatively large key size is required to ensure the RSA cryptosystem is computationally impossible to crack.

In some systems where the level of security is intolerable, the cryptography processing time required will become very significant. The situation becomes worse when the amount of data to be processed is huge. One of the typical example is the bank server system. Pure software implementation of RSA cryptography system, in this case, is too slow and insufficient to keep up with the computational demands of RSA cryptography processing. Hence, hardware implementation of RSA cryptosystem provides a practicable solution to the problem of the cryptography processing speed.

RSA algorithm using Binary Exponentiation consists of modular multiplication that required high computation cycles. Accelerating the modular multiplication operation will significantly help accelerating the whole RSA cryptography process. Thus, this project will focus on the hardware implementation to accelerate the Modular Multiplication in RSA coprocessor. Bit-level Montgomery Modular Multiplication algorithm that uses carry-saved adder for hardware parallel processing is implemented as the core of the RSA coprocessor to optimize the performance.

1.3 Objective of the Study

To implement and to improve the design of hardware-based 8192-bit RSA core which is able to handle RSA encryption and decryption efficiently. This is by implementing Bit-Level Montgomery Modular Multiplication using carry-saved adder to achieve hardware parallel processing.

1.4 Scope of the Study

Based on the outlined objectives above, available hardware and software resources, and the time frame allocated, this research project is narrowed down to the following scope of work.

1. The designed RSA core is able to handle 8192-bit RSA encryption and decryption correctly.
2. Synthesizable RTL code, Verilog HDL is used for the hardware implementation of the designed RSA core. The design has to be parameterized so that the coprocessor is reconfigurable for other key sizes, based on the required security level and the hardware resources constraints by targeted applications.
3. The logic functionality of the design need to be verified accurate in simulation environment. The simulation tool used is Altera ModelSim.

4. The RSA key pair generation of the RSA cryptosystem is not part of the scope of this project.

1.5 Report Organization

This project report is written in six chapters. The first chapter has introduced the background the problem as the motivation of this project. The problem statement, the objective and the scope of this project are clearly stated. The remaining chapters are organized as the followings:

1. Chapter 2 describes the theory part of public-key cryptography and RSA algorithm. In the same chapter, previous related works are discussed.
2. Chapter 3 explains the related algorithms used for this RSA hardware implementation.
3. Chapter 4 described in detailed of the translating the chosen algorithm into hardware description. Functional block diagrams and flow charts are used to assist the explanation of the design.
4. Chapter 5 shows the results obtained from the RSA hardware implementation simulation. This chapter also evaluate the area and performance of this project compare to the other related works.
5. Chapter 6 provides conclusion towards this project and gives recommendations for future works.

REFERENCES

Altera. Logic Array Blocks and Adaptive Logic Modules in Stratix IV Devices. In *Stratix IV Device Handbook*, chapter 2. Altera Corporation, 3.1 edition, 2011.

T. Blum and C. Paar. High-radix montgomery modular exponentiation on reconfigurable hardware. *Computers, IEEE Transactions on*, 50(7):759–764, 2001.

A. Daly and W. Marnane. Efficient architectures for implementing montgomery modular multiplication and rsa modular exponentiation on reconfigurable logic. In *Proceedings of the 2002 ACM/SIGDA tenth international symposium on Field-programmable gate arrays*, pages 40–49. ACM, 2002.

W. Diffie and M. E. Hellman. New directions in cryptography. *Information Theory, IEEE Transactions on*, 22(6):644–654, 1976.

T. ElGamal. A public key cryptosystem and a signature scheme based on discrete logarithms. In *Advances in cryptology*, pages 10–18. Springer, 1985.

J. Gantz and D. Reinsel. The digital universe in 2020: Big data, bigger digital shadows, and biggest growth in the far east. *IDC iView: IDC Analyze the Future*, 2007:1–16, 2012.

D. E. Knuth. The art of programming, vol. 2, semi-numerical algorithms, 1981.

N. Koblitz. Elliptic curve cryptosystems. *Mathematics of computation*, 48(177):203–209, 1987.

Y. Kong and Y. Lai. Low latency modular multiplication for public-key cryptosystems using a scalable array of parallel processing elements. In *Circuits and Systems (MWSCAS), 2013 IEEE 56th International Midwest Symposium on*, pages 1039–1042. IEEE, 2013.

V. Miller. Use of elliptic curves in cryptography. In *Advances in Cryptology-CRYPTO'85 Proceedings*, pages 417–426. Springer, 1986.

P. L. Montgomery. Modular multiplication without trial division. *Mathematics of computation*, 44(170):519–521, 1985.

A. Paniandi. *A hardware implementation of Rivest-Shamir-Adleman co-processor for resource constrained embedded systems*. PhD thesis, Universiti Teknologi Malaysia, Faculty of Electrical Engineering, 2006.

C. P. Rentería-Mejía, V. Trujillo-Olaya, and J. Velasco-Medina. Design of an 8192-bit rsa cryptoprocessor based on systolic architecture. In *Programmable Logic (SPL), 2012 VIII Southern Conference on*, pages 1–6. IEEE, 2012.

R. L. Rivest, A. Shamir, and L. Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Communications of the ACM*, 21(2):120–126, 1978.

A. C. Shantilal. A faster hardware implementation of rsa algorithm. *Oregon State University, Corvallis, Oregon*, 97331, 1993.