



Ain Shams University
Faculty of Engineering
Computer and Systems Engineering Department

Efficient Techniques for Cryptographic Algorithms Implementation

A Thesis

Submitted in partial fulfillment for the requirements of
Master of Science degree in Electrical Engineering

Submitted by:

Ahmad Mohammad Abdel-Fattah Ahmad

B.Sc. of Electrical Engineering
(Computer and Systems Engineering Department)
Ain Shams University, 2006

Supervised by:

**Prof. Dr. Hossam Mahmoud Ahmed
Fahmy**

Dr. Ayman Mohammad Bahaa Eldin

April 2010
Cairo

Statement

This dissertation is submitted to Ain Shams University for the degree of Master of Science in Electrical Engineering (Computer and Systems Engineering).

The work included in this thesis was carried out by the author at the Computer and Systems Engineering Department, Faculty of Engineering, Ain Shams University, Cairo, Egypt.

No part of this thesis was submitted for a degree or a qualification at any other university or institution.

Name: Ahmad Mohammad Abdel-Fattah Ahmad

Date: July 13, 2010

Acknowledgements

All praise is due to Allah, Most Merciful, the Lord of the Worlds, Who taught man what he knew not. I would like to thank God Almighty for bestowing upon me the chance, strength and ability to complete this work.

I wish to express my gratitude to my supervisors, Professor. Hossam Mahmoud Fahmy and Dr. Ayman Mohammad Bahaa Eldin for their exceptional guidance, encouragement, insightful thoughts and useful discussions. It was my great pleasure to work with them.

I am in no way capable of appropriately thanking my parents for their unconditional love and unlimited support.

Ahmad Mohammad Abdel-Fattah
Computer and Systems Engineering Department
Faculty of Engineering
Ain Shams University
Cairo, Egypt
2010

Abstract

The aim of this thesis is to develop a cryptographic accelerator module, focusing on public-key cryptographic algorithms.

Most public key cryptographic algorithms require heavy computational effort. This is due to the reliance on modular arithmetic over very large operands. Typical bit length of operands in public key ciphers is 1024 bits or even higher. Examples of such ciphers are RSA and Elliptic Curve Cryptography (ECC). Almost all public key algorithms involve modular multiplication and modular exponentiation. Such operations become very time-consuming when operating on large operands. Modular multiplication is the core of modular exponentiation.

Many algorithms have been developed to accelerate modular multiplication on general purpose processors. The common target of any accelerating algorithm is to avoid the ordinary pencil-and-paper steps. Many researchers managed to reduce both time and space complexities of modular multiplication.

However, considering embedded security devices such as smart cards and cryptographic tokens, the limited processing power of embedded microprocessor forces the need for more research to accelerate cryptographic algorithms on embedded platforms. The fast evolution of VLSI designs using Field Programmable Gate Arrays (FPGAs) and Application Specific Integrated Circuits (ASICs) has introduced further level of enhancement. The enhancement would be on the hardware level in the form of a cryptographic co-processor.

A cryptographic co-processor is a hardware accelerator that is specifically designed to perform complex cryptographic operations. The integration of such co-processor to a general purpose embedded CPU introduces an excellent platform for portable security devices.

In this thesis, a new hardware architecture for a modular multiplier is introduced. The design can be used to enhance the performance of all public key cryptosystems. The proposed architecture is based on the interleaved modular multiplication algorithm. It scores better timings (considering both operating frequency and total operation time) than the latest known designs to the author's knowledge. Further enhancements have been proposed to support parallel processing and different operand sizes. The proposed designs have been developed using a Hard-

ware Description Language (HDL). They have also been integrated to a soft processor core so that a complete platform is produced.

Contents

List of Figures	x
List of Tables	xii
List of Symbols	xiii
1 Introduction to Cryptology and Public Key Cryptography	1
1.1 Overview	1
1.2 Symmetric/Asymmetric Ciphers	2
1.3 Public Key Cryptography	2
1.3.1 Public Key Data Encryption and Decryption . . .	4
1.3.2 Digital Signatures	5
1.3.3 Key Exchange using PKCS	5
1.4 Challenges in Public Key Ciphers	6
1.5 Problem Statement and Thesis Objective	8
1.6 Methodology	8
1.7 Thesis Structure	9
2 Asymmetric Ciphers and Digital Signatures	11
2.1 DH Key Exchange Protocol	11
2.1.1 The DH Algorithm	12
2.2 RSA Cryptography	14
2.2.1 RSA Data Encryption and Decryption	14
2.2.2 RSA Key Generation	15

2.2.3	RSA Digital Signature Scheme	17
2.2.4	The Chinese Remainder Theorem	19
2.2.5	RSA Prime Factorization Problem	20
2.3	The Digital Signature Standard (DSS)	21
2.3.1	The DSS Approach	21
2.3.2	The Digital Signature Algorithm (DSA)	22
2.4	Elliptic Curve Cryptography (ECC)	26
2.4.1	Elliptic Curves over Prime Fields	26
2.4.2	Elliptic Curve Arithmetic based on Projective Co-ordinates	30
2.4.3	ECC Key Exchange Scheme	32
2.4.4	ECC Data Encryption and Decryption	33
2.4.5	Elliptic Curve Digital Signature Algorithm (ECDSA)	34
2.4.6	Elliptic Curve Discrete Logarithm Problem	36
2.4.7	ECC Computational Aspects	37
2.5	RSA vs. ECC	37
2.6	Summary	38
3	Modular Arithmetic	39
3.1	Modular Arithmetic	39
3.2	Modular Arithmetic Operations	41
3.3	Groups, Rings, and Fields	44
3.3.1	Groups	44
3.3.2	Rings	45
3.3.3	Integral Domain	45
3.3.4	Fields	46
3.4	Finite Fields of The Form $GF(P)$	47
3.4.1	Finite Fields of Order P	47
3.4.2	Testing for Primality	47
3.5	Large Integer Arithmetic	51
3.5.1	Large Integer Factorization Problem	51
3.5.2	Discrete Logarithm Problem	52
3.5.3	Multiple Precision Calculation	52
3.6	Modular Exponentiation	53
3.6.1	General Exponentiation	54
3.6.2	Fixed-base Exponentiation	54
3.6.3	Fixed-exponent Exponentiation	54

3.6.4	Square-and-Multiply Algorithm	54
3.7	Modular Multiplication	56
3.7.1	Montgomery Arithmetic	57
3.7.2	The Montgomery Multiplication Algorithm	58
3.7.3	Montgomery Exponentiation	59
3.7.4	Interleaved Modular Multiplication	61
3.8	Algorithms Analysis	62
3.9	Other Techniques	64
3.9.1	Barrett Reduction	65
3.9.2	Sliding Window Exponentiation	65
3.10	Summary	66
4	Hardware Implementation of Modular Multiplication	67
4.1	Binary Addition	67
4.1.1	Ripple Carry Adders (RCA)	68
4.1.2	Carry Look-Ahead Adders	69
4.1.3	Carry Save Adders (CSA)	71
4.2	Operands Caching using Look Up Tables	73
4.2.1	Montgomery Multiplication Algorithm	73
4.2.2	Interleaved Modular Multiplication Algorithm	74
4.3	Systolic Arrays and Parallel Computing	77
4.4	Sign Detection Technique	79
4.4.1	Operands Representation	80
4.4.2	Sign Detection Function	81
4.4.3	Correctness of the Sign Detection Technique	82
4.5	Summary	87
5	Proposed Modular Multiplier	88
5.1	The MIMM Algorithm	88
5.2	The MIMM Architecture	89
5.3	Hardware Modular Exponentiation	92
5.4	Modular Multiplication for ECC	95
5.5	Modular Multiplication for RSA using CRT	97
5.5.1	CMM Architecture	99
5.5.2	CMM Configuration Bits	100
5.5.3	Chaining Signals	101
5.6	Co-processor Integration	101

List of Figures

1.1	Symmetric Ciphers	3
1.2	Asymmetric Ciphers	3
1.3	Public Key Encryption and Decryption	4
1.4	Message Authentication	6
2.1	RSA Based Digital Signature	22
2.2	DSA Based Digital Signature	22
2.3	DSA Signature Generation	24
2.4	DSA Signature Verification	25
2.5	An Example Elliptic Curve	27
2.6	Geometric Representation of ECC Point Addition	29
3.1	Geometric View of Arithmetic Modulo 12	40
3.2	Groups, Rings, Integral Domains, and Fields	46
3.3	Square and Multiply Algorithm	56
3.4	Bit-level Montgomery Multiplication	59
3.5	Word-level Montgomery Multiplication	60
3.6	Montgomery Exponentiation Algorithm	61
3.7	Interleaved Modular Multiplication	62
3.8	Number of Operations per a Modular Multiplication	64
3.9	Number of Operations per a Modular Exponentiation	65
4.1	One Bit Full Adder	68
4.2	Ripple Carry Adder	69
4.3	n-bits Carry Save Adder	72
4.4	Montgomery Multiplication Using Look-up Tables	74

4.5	Redundant Interleaved Modular Multiplication	75
4.6	Optimized Interleaved Modular Multiplication	77
4.7	Optimized Interleaved Modular Multiplier	77
4.8	Example Systolic Array	78
4.9	Sign Estimation Format	81
5.1	MIMM Algorithm	89
5.2	MIMM as Black Box	90
5.3	Internal MIMM Design	91
5.4	MIMM State Machine	92
5.5	Processor-Multiplier Core Communication	94
5.6	Modular Exponentiation Accelerator	95
5.7	ECC Modular Multiplier	97
5.8	CMM Architecture	99
5.9	Avalon-MM Interface	104
5.10	ECC Multiplier with Avalon Interface	105
5.11	Custom Instruction	107
5.12	Custom Instruction Ports	108
5.13	Combinational Custom Instruction Operation	108
5.14	Multi-cycle Custom Instruction Operation	110
5.15	MIMM Test Bench	114
5.16	Custom Instruction Test Bench	115
5.17	Test Bench State Machine	116
5.18	Custom Peripheral Test Bench	117

List of Tables

2.1	Computational Effort for Cryptanalysis of RSA and ECC	37
3.1	Multiplication Modulo 8	43
3.2	Addition in GF(7)	48
3.3	Multiplication in GF(7)	48
3.4	Miller Rabin Primality Test	50
4.1	Sign for Negative 2-bit Windows	83
4.2	Sign for Negative 3-bit Windows	83
4.3	Sign for Negative 4-bit Windows	84
4.4	Sign Decision versus Windows Values	86
5.1	MIMM States	93
5.2	Memory Map for Avalon ECC Multiplier	106
5.3	Custom Instruction Types	109
5.4	ECC Multiplier Instruction	112
5.5	Device Utilization for Modular Multipliers	119
5.6	Max. Frequency for Modular Multiplier	119
5.7	Absolute Time (μs) per an Operation	120
5.8	Implementation Results for CMM	120
5.9	Results for Other CMM Versions	121
5.10	Implementation Results for ECC Modular Multiplier	121
5.11	Timing Results for ECC Modular Multiplier	122

List of Symbols

AES	Advanced Encryption Standard
ALU	Arithmetic and Logic Unit
ASIC	Application Specific Integrated Circuit
CLA	Carry Lookahead Adder
CMM	Chained Modular Multiplier
CPU	Central Processing Unit
CRT	Chinese Remainder Theorem
CSA	Carry Save Adder
DES	Data Encryption Standard
DH	Diffie-Hellman
DMIPS	Dhrystone Million Instruction per Second
DPU	Data Processing Unit
DSA	Digital Signature Algorithm
DSS	Digital Signature Standard
ECC	Elliptic Curve Cryptography
ECDLP	Elliptic Curve Discrete Logarithm Problem
ECDSA	Elliptic Curve Digital Signature Algorithm
FIPS	Federal Information Processing Standard
FPGA	Field Programmable Gate Array
FSM	Finite State Machine
GCD	Greatest Common Divisor
HDL	Hardware Description Language
IMM	Interleaved Modular Multiplication
KDC	Key Distribution Center

LHS	Left Hand Side
MIMM	Modified Interleaved Modular Multiplication
MSB	Most Significant Bit
NIST	National Institute of Standards and Technology
PCB	Printed Circuit Board
PKCS	Public Key Cryptography Standard
PKI	Public Key Infrastructure
RHS	Right Hand Side
RCA	Ripple Carry Adder
RNS	Residue Number System
RSA	Rivest-Shamir-Adelman
SHA	Secure Hash Algorithm
UUT	Unit Under Test

Chapter 1

Introduction to Cryptology and Public Key Cryptography

This chapter discusses the basic principles of public key cryptography, and the difference between symmetric and asymmetric (public key) cryptographic algorithms. It also points out the appearing challenges when implementing a public key cryptosystem. The chapter also clarifies the objective of this thesis. It then presents the methodology followed during the research. The chapter ends with an illustration of the thesis structure.

1.1 Overview

Cryptology is defined as the study of techniques for ensuring the secrecy and authenticity of information [1]. The study of cryptology is divided into two main categories: Cryptography and Cryptanalysis. Cryptography involves methods and techniques to achieve the secrecy of data. Such secrecy should be accomplished using an encryption/decryption algorithm (cipher). Cryptanalysis is the study of different ways to attack a cryptographic algorithm in order to break the security of information. Cryptanalysis provides methodologies for measuring how strong a cipher is. In this work, a focus is made on cryptography.