

An RNS-Enabled Microprocessor for Public Key Cryptography

Zhining Lim

B.E.(Comp. Sys.)(Hons.)



A THESIS SUBMITTED IN FULFILMENT OF THE
REQUIREMENTS FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY

SCHOOL OF ELECTRICAL AND ELECTRONIC ENGINEERING
THE UNIVERSITY OF ADELAIDE · AUSTRALIA

February 2010



Copyright © 2010

Zhining Lim

All rights reserved

CONTENTS

List of Figures	v
List of Tables	viii
List of Algorithms	ix
Abstract	xi
Declaration of Originality	xiii
Acknowledgements	xv
Publications	xvii
List of Abbreviations	xix
List of Principal Symbols	xxi
1 Introduction	1
2 Background	5
2.1 The Residue Number System	5
2.1.1 Graphical RNS Representation	7
2.1.2 Existing RNS Implementations	8
2.2 Public Key Cryptography	10
2.2.1 The RSA Algorithm	13
2.2.2 Elliptic Curve Cryptosystems	14
2.2.3 Side-Channel Attacks	17
2.3 Smart Cards	19
2.4 The Xtensa LX2.1 Processor	22

3	Project Design	25
3.1	Hypotheses	25
3.2	Method	26
3.2.1	Smart Card Specifications	27
3.2.2	Building the \mathbb{Z} -Core	27
3.2.3	Public Key Cryptosystem Implementations	28
3.2.4	Measuring Area and Power	30
4	The \mathbb{Z}-Core	33
4.1	Memory and Register Structure	33
4.2	\mathbb{Z} -Core Operations	36
4.2.1	Memory Access Operations	38
4.2.2	Modular Arithmetic Operations	40
4.2.3	Miscellaneous Operations	43
4.3	Area and Power Measurements	46
4.3.1	Area Measurements	47
4.3.2	Power Measurements	48
4.4	Conclusion	49
5	The RSA Cryptosystem	51
5.1	Modular Exponentiation	51
5.2	Modular Reduction	56
5.3	RNS Modular Multiplication	59
5.3.1	Kawamura's Base Extension	61
5.3.2	Bajard's Base Extension	63
5.3.3	Hybrid Base Extension	65
5.4	Multiprecision Modular Multiplication	66
5.5	Chinese Remainder Theorem	69
5.6	Implementation	73
5.6.1	Residue Number System	73
5.6.2	Multiprecision Baseline	76

5.6.3	Chinese Remainder Theorem	77
5.7	Results	78
5.7.1	Modular Multiplication	79
5.7.2	Modular Exponentiation	80
5.7.3	Cache Considerations	82
5.7.4	Memory Usage	85
5.7.5	Power Consumption	86
5.8	Discussion	88
5.8.1	Performance	89
5.8.2	Area	92
5.8.3	Power	92
5.8.4	Comparisons	93
5.9	Conclusion	94
6	The Elliptic Curve Digital Signature Algorithm	97
6.1	Elliptic Curve Digital Signature Algorithm	97
6.2	Elliptic Curve Point Multiplication in the RNS	99
6.3	Elliptic Curve Point Arithmetic	101
6.3.1	Projective Montgomery	103
6.3.2	Chudnovsky Jacobian	105
6.4	Multiplicative Inverse	107
6.5	Results	114
6.5.1	ECDSA Operations	115
6.5.2	Cache Considerations	117
6.5.3	Memory Usage	118
6.5.4	Power Consumption	118
6.6	Discussion	119
6.6.1	Performance	119
6.6.2	Area	122
6.6.3	Power	122

6.6.4	Comparisons	122
6.7	Conclusion	126
7	Conclusion	129
7.1	Implementations on the \mathbb{Z} -Core	129
7.2	Limitations of the \mathbb{Z} -Core	130
7.3	Future Work	131
A	RNS TIE Code	135
A.1	Register File and States	135
A.2	Functions	135
A.3	Operations	136
A.3.1	Memory Access Operations	136
A.3.2	Modular Arithmetic Operations	139
A.3.3	Miscellaneous Operations	139
B	RNS Modular Multiplication Code	143
C	Multiprecision Modular Multiplication Code	147
	Bibliography	151

LIST OF FIGURES

2.1	RNS diagram depicting channel multiplication of G by H	8
2.2	Symbols used in the RNS diagrams.	9
2.3	ECC point addition of P and Q	16
2.4	ECC point doubling of P	16
4.1	First attempt at the RNS channels for the \mathbb{Z} -Core.	34
4.2	Second attempt at the RNS channels for the \mathbb{Z} -Core.	35
4.3	Final hardware structure of the RNS channels for the \mathbb{Z} -Core.	36
4.4	Load-store overhead for sequential implementation options.	37
4.5	Inner loop of an RNS CRT base extension.	45
4.6	RNS MRS base extension for one channel.	46
4.7	Area usage of \mathbb{Z} -Core TIE hardware.	47
4.8	Power usage of the baseline and \mathbb{Z} -Core.	49
5.1	RNS Montgomery multiplication scheme.	60
5.2	Kawamura's RNS Montgomery modular multiplication algorithm.	62
5.3	Bajard's RNS Montgomery modular multiplication algorithm.	63
5.4	Hybrid RNS Montgomery modular multiplication algorithm.	66
5.5	Multiprecision multiplication.	70
5.6	Comparison of modular multiplication times for varying input lengths.	81
5.7	Data cache stall cycles for 1,024-bit RNS exponentiation.	83
5.8	Proportion of data cache stall cycles for varying input length.	84
5.9	Proportion of data cache stall cycles for varying cache and input size.	85
6.1	Simplified CRT base extension for multiplicative inverse.	112
6.2	Simplified MRS base extension for multiplicative inverse.	114

LIST OF TABLES

2.1	Effective key-size of common public key cryptology methods.	12
2.2	Voltage, current, and power for major smart card standards.	20
2.3	ARM SecurCore family characteristics.	21
2.4	1,024-bit RSA signature generation on the ARM SecurCore.	22
3.1	Smart card specifications to be met by the \mathbb{Z} -Core.	28
4.1	Memory access operations.	39
4.2	Modular arithmetic operations.	41
4.3	Miscellaneous operations.	44
4.4	Synthesis areas for the \mathbb{Z} -Core.	48
5.1	RNS Montgomery modular multiplication operation counts.	67
5.2	Modular multiplication operation counts from implementations.	73
5.3	Clock cycles required for a 1,024-bit modular multiplication.	80
5.4	Clock cycles required for a 1,024-bit modular exponentiation.	82
5.5	Memory used in 1,024-bit RSA implementations.	86
5.6	Cache area and power.	87
5.7	Power for cache and memory in 1,024-bit RSA implementations.	88
5.8	Best RSA signature generation results on the two platforms.	90
5.9	Comparison of RSA implementations.	93
5.10	Comparison of CRT RSA implementations.	94
6.1	Modular reductions for one doubling and one addition in the RNS. . .	103
6.2	Relations for the least significant bit of the subtraction result.	113
6.3	Elliptic curve and ECDSA operations for 192-bit points in the RNS. . .	116
6.4	\mathbb{Z} -Core RNS signature generation results for RSA and ECDSA.	120
6.5	Elliptic curve point multiplication comparison.	124

LIST OF ALGORITHMS

4.1	Crandall's modular reduction algorithm.	42
5.1	Square-and-multiply-always modular exponentiation.	53
5.2	Exponent splitting for modular exponentiation.	54
5.3	Modular exponentiation with exponent blinding.	57
5.4	CIOS multiprecision Montgomery modular multiplication.	68
5.5	Modular inverse.	70
6.1	Elliptic curve digital signature generation.	98
6.2	Elliptic curve digital signature verification.	98
6.3	Modified Montgomery ladder elliptic curve multiplication.	100
6.4	Variant of Shamir's double ladder.	102
6.5	RNS projective Montgomery addition.	106
6.6	RNS projective Montgomery doubling.	106
6.7	RNS Chudnovsky Jacobian addition.	108
6.8	RNS Chudnovsky Jacobian doubling.	108
6.9	Right-shift algorithm for the classical modular inverse.	110

ABSTRACT

Mobile computing platforms must be secure, fast, small, and power efficient. While public key cryptography methods provide the framework for secure communications, their algorithms tend to be computationally complex and hence need some form of hardware acceleration. This thesis investigates an alternative implementation of public key cryptology algorithms on a microprocessor that has been augmented with hardware support for the residue number system (RNS).

The RNS promises efficient arithmetic by replacing the long integers used in public key cryptology with sets of smaller independent numbers. Doing so allows flexibility in instruction scheduling and reduces carry propagation delays. Past RNS public key cryptosystem implementations used a parallel architecture. Although well-suited to the RNS, a parallel architecture is not necessarily power efficient nor small since it requires several copies of the RNS hardware. The microprocessor implementation in this thesis has a novel sequential architecture that has modest area and power consumption because it uses just one set of the RNS hardware.

The RNS-enhanced microprocessor, the \mathbb{Z} -Core, is based on the small, low-power Xtensa LX2.1 core from Tensilica, Inc. It was augmented with RNS hardware specified using the Tensilica Instruction Extension language. Unlike other cryptographic accelerators, these RNS enhancements are useful for a variety of cryptosystems. The \mathbb{Z} -Core was used for implementations of RSA and the elliptic curve digital signature algorithm. The RNS implementations on the \mathbb{Z} -Core outperformed equivalent implementations that use the normal multiprecision methods. The area and power consumption of the \mathbb{Z} -Core were within the GSM 11.18 1.8-V SIM specifications for smart cards when running the two algorithms, hence it satisfied the requirements of mobile computing.

DECLARATION OF ORIGINALITY

This work contains no material which has been accepted for the award of any other degree or diploma in any university or other tertiary institution to Zhining Lim and, to the best of my knowledge and belief, contains no material previously published or written by another person, except where due reference has been made in the text.

I give consent to this copy of the thesis, when deposited in the University Library, being available for loan, photocopying, and dissemination through the library digital thesis collection, subject to the provisions of the Copyright Act 1968.

I also give permission for the digital version of my thesis to be made available on the web, via the University's digital research repository, the Library catalogue, the Australasian Digital Thesis Program (ADTP) and also through web search engines, unless permission has been granted by the University to restrict access for a period of time.

Zhining Lim

8 February 2010

ACKNOWLEDGEMENTS

My deepest gratitude goes to my supervisor, Dr Braden Phillips. I would like to thank him for his continued support, encouragement, and helpful ideas along the way. I would also like to thank my co-supervisor, Associate Professor Michael Liebelt, for his guidance in the project.

This project would not have been possible without the generosity of Tensilica, Inc. in Santa Clara, California, USA. Under their University Program, I was given extensive access to their Xtensa LX2.1 microprocessor core. Their RTL licence provided the capabilities to run area and power measurements using Synopsys tools. I would not have been able to explore the RNS architectural enhancements nor collect the results as easily and painlessly without the Tensilica tools.

This project has been supported by the Australian Research Council's Discovery Projects funding scheme (project number DP0557582). My research candidature was supported by an Australian Postgraduate Award and by the Frank Perry Supplementary Scholarship from the University of Adelaide. In addition, I gratefully acknowledge the IEEE South Australian Section for covering some of my travel expenses through their Travel Award.

Finally, but in no means the least, I would like to thank my family for their support and love.

Zhining Lim

8 February 2010

PUBLICATIONS

- LIM, Z., PHILLIPS, B. J., AND LIEBELT, M. 2009. Elliptic Curve Digital Signature Algorithm over $GF(p)$ on a Residue Number System Enabled Microprocessor. In *IEEE Region 10 Conference TENCON 2009*. Singapore.
- LIM, Z., AND PHILLIPS, B. J. 2007. An RNS-Enhanced Microprocessor Implementation of Public Key Cryptography. In *41st Asilomar Conference on Signals, Systems, and Computers*. Pacific Grove, California, USA, 1430–1434.
- LIM, Z., AND PHILLIPS, B. J. 2007. An RNS-Enabled Architecture on the Tensilica Xtensa LX Processor. Tech. Rep. CHIPTEC-07-01. Centre for High Performance Integrated Technologies and Systems, The University of Adelaide, Adelaide, Australia.

LIST OF ABBREVIATIONS

ANSI	American National Standards Institute
CIOS	Coarsely Integrated Operand Scanning
CISC	Complex Instruction Set Computer
CRT	Chinese Remainder Theorem
DPA	Differential Power Analysis
ECC	Elliptic Curve Cryptography or Cryptosystem
ECDSA	Elliptic Curve Digital Signature Algorithm
FIPS	Federal Information Processing Standards
FPGA	Field Programmable Gate Array
GSM	Global System for Mobile communications
IEC	International Electrotechnical Commission
ISO	International Organisation for Standardisation
MRS	Mixed Radix System
NIST	National Institute of Standards and Technology
RISC	Reduced Instruction Set Computer
RNS	Residue Number System
RTL	Register Transfer Level
RSA	Rivest-Shamir-Adleman Public Key Cryptosystem
SECG	Standards for Efficient Cryptography Group
SPA	Simple Power Analysis
SRAM	Static RAM
TIE	Tensilica Instruction Extension language

TSMC Taiwan Semiconductor Manufacturing Company

LIST OF PRINCIPAL SYMBOLS

\vee	Bit-wise OR
\wedge	Bit-wise AND
\leftarrow	Base extension
\mathcal{O}	ECC point at infinity
a	32-bit Xtensa address register, or its contents
A	64-bit user-defined register, or its contents
A_L	Lower 32-bits of 64-bit A
A_U	Upper 32-bits of 64-bit A
\mathcal{B}_A	RNS base A
\mathcal{B}_{AUB}	RNS base containing channel moduli of base A and base B
(C, S)	Product of two numbers separated into two words, S and C
G_i	Channel i of RNS form of G
$G \text{ mmod } p$	Montgomery modular reduction of G modulo p
$\langle G \rangle_p$	RNS value G reduced modulo p
$\langle G \rangle_{\mathcal{B}_A}$	RNS value G reduced modulo all moduli in RNS base A
\tilde{G}	RNS value G with corrective Montgomery factor
k_i	i th bit or word of k , depending on the context
m_i	Modulus of RNS channel i
m_{A_i}	Modulus of RNS channel i of \mathcal{B}_A
M	Dynamic range of an RNS base
M_A	Dynamic range of \mathcal{B}_A
M_{A_i}	Dynamic range of \mathcal{B}_A divided by modulus of channel i in \mathcal{B}_A

$\text{MEM}_{32} [a]$	32-bit memory access at memory address a
$\text{MEM}_{64} [a]$	64-bit memory access at memory address a
$R[i]$	i th element of array R
x_{dec}	A number x in decimal form or base 10
x_{hex}	A number x in hexadecimal form or base 16
x_G	x -coordinate of ECC point G in affine form
X_G	x -coordinate of ECC point G in a projective form
$\langle x \rangle_p$	Modular reduction of x modulo p
$\langle x^{-1} \rangle_p$	Multiplicative inverse of x modulo p