Universitat Autònoma de Barcelona

ECC-based Authentication in Constrained Devices: A Secure and Efficient Hardware Implementation

Sadiel de la Fé Siverio

Universitat Autònoma de Barcelona

Departament de Microelectrònica i Sistemes Electrònics

ECC-based Authentication in Constrained Devices: A Secure and Efficient Hardware Implementation

A Thesis Submitted for the Degree of
PhD in Electrical and Telecommunications Engineering

Author:
Sadiel de la Fé Siverio

Director:
Carles Ferrer Ramis

Barcelona, 2021

DECLARATION

I hereby declare that I am the sole author of this thesis and that the work contained is original, except where specific reference is made to the work of others.

Barcelona, September 24, 2021

[illegible]

____

Sadiel de la Fé Siverio

ECC-based Authentication in Constrained Devices: A Secure and Efficient Hardware Implementation

A Thesis Submitted for the Degree of
PhD in Electrical and Telecommunications Engineering

PhD candidate: Sadiel de la Fé Siverio
Signature

[illegible]

Supervisor: Carles Ferrer Ramis
Signature

Universitat Autònoma de Barcelona
Department of Microelectronics and Electronic Systems
Edifici Q, C/Sitges,
08193, Cerdanyola del Valles, Spain

September 24, 2021

A mi familia

Acknowledgment

This work would have never been done without the help of those people to whom I would like to express herein my deepest gratitude. Many thanks to those that are impossible to mention here and somehow helped me to achieve this goal.

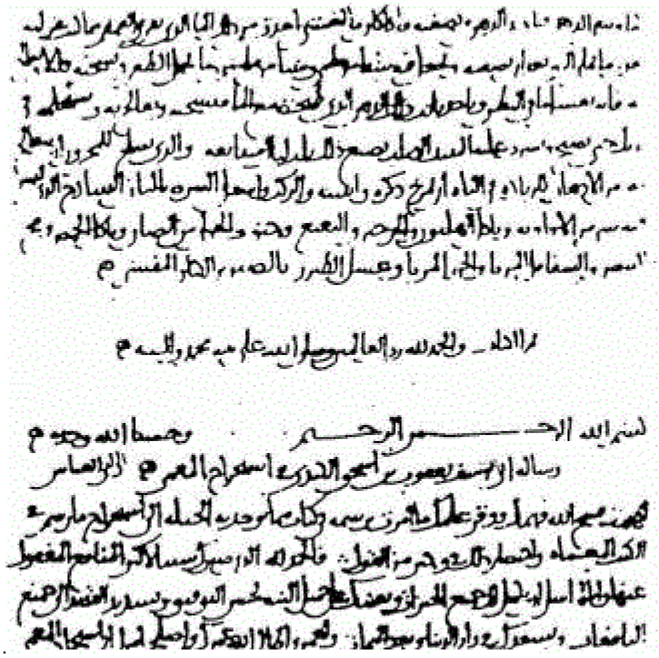
In the first place I would like to express my sincere gratitude to Ricardo Chaves, Pedro Matutino and Juvenal Araujo, from the Instituto Superior Técnico de Lisboa for their invaluable guidance and support during the development of a core part of this thesis.

My special thanks to the supervisor of this thesis: Carles Ferrer, for his guidance and for helping me out to conduct an autonomous research. All my gratefulness to professors Lluis Ribes, Elena Valderrama, Jordi Aguiló, Paco Serra and the secretarial staff from the Microelectronics and Systems Dpt. of UAB, for their availability and help.

All my gratitude to Erica Tena and Prof. Antonio J. Acosta from the University of Sevilla, and also to Prof. Bo-Yeon Sim and Dong-Guk Han from the Kookmin University in Seoul, for their collaboration in valuable experimentations for this work.

Thanks to my colleagues of the PhD: Adriana, Natalie, Biruk for their support and especially to Lu Wang for her invaluable help.

My deepest gratitude to my family, specially to my parents and my wife for their patience and encouragement throughout all these years.



[illegible]

[illegible] والحمد لله رب العالمين [illegible]

بسم الله الرحمن الرحيم   وحسبنا الله وحده

رسالة ابي يوسف يعقوب بن اسحق الكندي في استخراج المعمى [illegible]

[illegible]

A Manuscript on Deciphering Cryptographic Messages.

Al-Kindi, 9th Century

Abstract

The proliferation of connected devices has increased in the last few years in the form of wearable personal gadgets or sensors. Into the smart cities and Internet of Things (IoT) concepts the sensors are intended to acquire measurements from different sources to provide an improved management. These facts imply that many connected devices today can manipulate critical or sensitive data from infrastructures or persons.

On the other hand, the proliferation of cyberattacks has grown, according to cybersecurity companies. In the scientific literature and also in the news, one can see examples of exploited vulnerabilities in devices that handle sensitive data. The weak or the lack of authentication is a common issue among many devices, allowing a potential attacker to log into the device and even launch a higher level attack. The lack of physical protection of cryptographic implementations make some devices even more vulnerable.

This thesis aims to provide a hardware implementation of a suitable authentication scheme to fit in low-power devices. The work focuses on the optimization and hardening of the critical cryptographic operations involved in the authentication. The theoretical and experimental implementation provided show that an efficient authentication scheme can be embedded in low-power connected devices, where some of the more hazardous physical attacks are also prevented.

Resumen

La proliferación de dispositivos conectados ha aumentado en los últimos años en forma de aparatos personales o sensores. En lo que se conoce hoy como Ciudades Inteligentes e Internet de las Cosas, los sensores están destinados a la adquisición de medidas de diferentes fuentes para proporcionar una mejor gestión y control de los recursos. Estos implican que muchos de estos dispositivos actualmente pueden manipular datos críticos o sensibles para infraestructuras o personas.

El número de ciberataques ha crecido, según varias empresas de ciberseguridad. La literatura científica y las noticias reflejan ejemplos de vulnerabilidades en dispositivos que gestionan datos sensibles. La falta de esquemas de autenticación o las implementaciones débiles son un problema común entre muchos dispositivos, ya que permite a un atacante potencial iniciar sesión en el dispositivo e incluso lanzar un ataque a nivel de red. La falta de protección física de los algoritmos criptográficos hace que algunos dispositivos sean aún más vulnerables.

El objetivo de esta tesis es proporcionar la implementación hardware de un esquema de autenticación adecuado para dispositivos limitados en potencia. El trabajo se centra en la optimización y la robustez de las operaciones criptográficas críticas implicadas en la autenticación. El análisis teórico y la evaluación experimental desarrollados demuestran que es posible implementar un esquema de autenticación eficiente en dispositivos conectados de baja potencia, donde también se evitan algunos de los ataques físicos.

Resum

La proliferació de dispositius connectats ha augmentat en els darrers anys en forma de dispositius o sensors personals. En els conceptes de Smart Cities i IoT, els sensors estan destinats a adquirir mesures de diferents fonts per proporcionar una gestió millorada. Aquests fets impliquen que molts dispositius connectats actualment poden manipular dades crítiques o sensibles per a infraestructures o persones.

D’altra banda, la proliferació de ciberatacs ha crescut, segons les empreses de ciberseguretat. A la literatura científica i també a les notícies, es poden veure exemples de vulnerabilitats explotades en dispositius que gestionen dades sensibles. La dèbil o la manca d’autenticació és un problema comú entre molts dispositius, ja que permet a un atacant potencial iniciar sessió al dispositiu i fins i tot iniciar un atac de nivell superior. La manca de protecció física dels algorismes criptogràfics fa que alguns dispositius siguin encara més vulnerables.

Aquesta tesi té com a objectiu proporcionar una implementació de maquinari d'un esquema d'autenticació adequat per adaptar-se a dispositius de baixa potència. El treball se centra en l'optimització i l'enduriment de les operacions criptogràfiques crítiques implicades en l'autenticació. La implementació teòrica i experimental proporcionada mostra que es pot encastar un esquema d’autenticació eficient en dispositius connectats de baixa potència, on també s’eviten alguns dels atacs físics més perillosos.

Contents

ACRONYMS

AES	Advanced Encryption Algorithm
BEEA	Binary Extended Euclidean Algorithm
DSA	Digital Signature Algorithm
DBLU	Point Doubling (with update)
DPA	Differential Power Analysis
ECC	Elliptic Curve Cryptography
ECDSA	Elliptic Curve Digital Signature Algorithm
FIFO	First In First Out
FLT	Fermat's Little Theorem
GCD	Greatest Common Divisor
GPIO	General Purpose Input Output
HSM	Hardware Secure Module
IBS	Id-based Signature Scheme
IoT	Internet of Things
LSB	Least Significant Bit
MA-PADD	Mixed-Affine Point Addition
MPM	Multiple point multiplication
NAF	Non Adjacent Form
PD	Point Doublings
PM	Point multiplication
RF	Radio Frequency
RNS	Residue Number System
RSA	Rivest-Shamir-Adleman Algorithm

SCA	Side Channel Attack
SPA	Simple Power Attack
TA	Template Attacks
UADD	Unified Point Addition
ZADD	Co-Z Point Addition
ZADDC	Co-Z Conjugate Point Addition
ZADDU	Co-Z Point Addition (with point update)

List of figures

List of tables

Introduction

1.1 Motivation

Smart city concept makes reference to sustainability in terms of optimization of resources, to give response to the increasing population in urban cities. In that sense, the European Union works towards a strategy to incorporate more intelligence in their cities through a more efficient resources management, in parallel to the population growth [1]. Some of the big companies already provide solutions to approach the Smart City concept from the data connectivity, mobility or energy point of view [2] [3].

Among the main subsystems a smart city envisages are: Distributed electricity generation based on multiple microgeneration points, Smart grids of interconnected data networks, Smart metering for the consumption measurement of water and electricity, Smart buildings with automated systems for the optimum use of energy, and Smart sensors (recently incorporated in the IoT concept) to collect the multitude of data generated around this Smart ecosystem. In addition, portable personal devices contribute with mobility data, which may be used to optimize the public transportation system while reducing CO_2 concentrations.

The use of connected sensors or portable devices gathering data from critical infrastructures or persons has a few of inherent risks [4]. The Open Web Application Security Project (OWASP) defined a Top 10 list of the most important cybersecurity threats in IoT for 2018 [5]. Some of the threats are based on unauthorized access to sensitive data and system management, which are related to weak authentication mechanisms. Also, the lack of physical hardening measures is highlighted as it allows the attacker to take control of the device, thus having the possibility to launch a high-level network attack from inside.

Some real life examples of vulnerabilities have been recently reported. Bluetooth, the well-known wireless technology for short range communications, was found vulnerable in [6], due to an exploitable issue with non-authenticated cryptographic keys. In [7] the authors discovered severe vulnerabilities in the communication of some kids-oriented smartwatches. The researchers found that some devices did not use any authentication or encryption scheme and in some case, instead of a proper authentication, a device identification was implemented through the IMEI (International Mobile Equipment Identity), which can be easily retrieved. In [8] a tampering attack is described against a supposedly high secure device. Through a physical access to the HSM (Hardware Secure Module), the researchers had access to private objects and they even managed to disable some cryptographic mechanism.

In view of the previous facts, one may reasonably suspect that a large number of devices lack of adequate security measures. An exploitable flaw on some of them may cause a high impact on infrastructures or in human beings. The cryptographic algorithms implemented to provide a certain protection level are not vulnerable by themselves, but a physical access to the device may lead to disable such protection. Thus, high-level security mechanisms along with tamper-proof countermeasures should be considered to guarantee the safety of the previously mentioned Smart city subsystems. Furthermore, some of those devices are typically battery-powered and constrained in computational resources, therefore, the cryptographic implementations should consider those limitations.

Objective: The main objective of this thesis is to provide a hardware implementation for secure and efficient authentication of low-power devices. In order to achieve the goal, this work focuses on the optimization and hardening of the most critical cryptographic operations involved in the authentication.

The research in this thesis involved the study of some authentication schemes with an adequate efficiency for being implemented in low-power devices. The deep analysis of the algorithms and their mathematical operations lead to an algorithmic optimization that improves one stage of the authentication process. The selection of the authentication schemes for the implementation also included an analysis considering their compatibility with the hardware platform used. The said platform includes a FPGA-based RNS processor which was selected because its implementation allows both efficient computation and intrinsic protection against some physical attacks. The implementation of the critical cryptographic operations involved in the authentication schemes was conducted in the hardware platform, while its performance was evaluated and further compared with similar works.

Additionally, the modular inversion operation was independently analyzed because of its importance in the authentication process. From the analysis, two physical attacks and the respective countermeasures were proposed by the first time against two modular inversion operations.

1.2 Background

1.2.1 Authentication

Authentication is the mechanism through which a message, a user or a device proves its legitimacy. Authentication schemes (a.k.a Signature schemes) might the most important among the cryptographic methods in a Smart sensors subsystem.

An efficient method to authenticate in a point-to-point or point-to-multipoint communication is the use of message authentication codes (MAC). If all the network elements share a predefined secret, a symmetric cryptographic primitive can be used to generate MACs at a very low cost [9-10]. Thus, a node sends a message along with a MAC, and a receiver employs the symmetric key to evaluate whether message-MAC combination holds, in which case, the message is accepted. However, in events like network scaling, eventual external devices log in to a network, or broadcast messages, the MAC based on shared secret is impractical. Moreover, to share a secret among all the devices increases the risk of one of them being compromised.

RSA (Rivest, Shamir, Adleman – its creators) was devised in 1977 as a public-key scheme, and is arguably the most widespread method to sign data [11]. Its strength relies in that it is impractical to decompose large integers. In modern systems, RSA secret parameters are typically in a range from 1024 bits to 4096 bits. However, the first claim of RSA was to be a method in which a (public) key could be exposed without losing the system security. To send a signed message, a sender first publishes its public key, and then uses its private key to sign the message. A receiver uses the sender's public key to verify the signature for that specific message, and only if it holds the communication is accepted.

What about the legitimacy of the public key?

If any public key is accepted, then any device is theoretically allowed to send data through a network. To void it, a public key is required to have a certificate that proves its legitimacy to others, like the standard in [12]. Thus, in a secure communication, a receiver must have access to the sender's public key and its correspondent certificate. In a network of constrained devices, such complexity is not desirable.

To overcome that issue, a method was proposed to authenticate via a certificate-less public-key cryptosystem [13]. Actually, Shamir was the first to propose an identification-based authentication, which opened the door for more efficient implementations in low-power devices. This topic will be addressed more exhaustively in the next chapter.

1.2.2 Elliptic Curve Cryptography

The Elliptic Curves Cryptography (ECC) is based on the use of certain curves defined by a set of points (x, y) over a $GF(p)$, which are solutions to the equation

$$y^2 = x^3 + ax + b \quad (1.1)$$

along with a point called point at the infinity. The point operations like addition, subtraction and multiplication follow certain rules in this kind of curves and are geometrically defined [14-15].

A point $P \in E(\mathrm{F}_p)$, with order n, is a generator of an ECC curve, which can be defined through

$$E(\mathrm{F}_p) = \{\infty, P, 2P, 3P, \ldots, (n-1)P\} \quad (1.2)$$

In a public-key cryptosystem based on ECC, the large prime p, the curve equation (1.1), and the generator point P are public parameters. A private key k is randomly selected in the range $[1, n-1]$ and the public key Q is obtained through the Point Multiplication (PM)

$$Q = kP \quad (1.3)$$

The Discrete Logarithm Problem (DLP) defined the intractability of determining x from $y = g^x \, mod \, p$ under certain conditions [16]. DLP can be extended to ECC and is known as the Elliptic Curve Discrete Logarithm Problem (ECDLP). Based on the ECDLP, the recovery of k in (1.3) is intractable, even having all the public parameters including the public key Q [14-15].

The PM is performed following the old Booth's algorithm employing doublings and additions [17]. The point doublings and additions in ECC imply divisions in both cases, if the point representation is given in affine coordinates (x, y). Divisions are costly operations in digital systems, thus, should be avoided. A representation of the ECC points in projective coordinates is preferred, as divisions are no longer needed [14].

A projective point is represented as $(X{:}\,Y{:}\,Z)$. A projective point has representatives in the form $(\lambda^c X, \lambda^d Y, \lambda Z)$, where $c = d = 1$ (standard coordinates) or $c = 2\ and\ d = 3$ (Jacobian coordinates) and λ is an integer. It means every point $(X', Y', Z') \equiv (X'', Y'', Z'')$ if both are representatives of a same point $(X{:}\,Y{:}\,Z)$. The affine coordinate (x, y) is then computed as $(X/\lambda^c, Y/\lambda^d, 1)$ being $Z = 1/\lambda$ [14].

Today is widely accepted that to achieve an adequate security level, an ECC cryptosystem needs, at least a 224-bits key while RSA needs a 2048-bits key [18-19]. Such metrics make ECC more efficient than RSA for the implementation in constrained devices.

1.3 Thesis outline

This thesis is divided into six chapters. In the current chapter, an introduction is provided to define the context of this work and highlight one specific problem and the suggested solution. Also, a background is given on the main topics around the work: Authentication and Elliptic Curve Cryptography (ECC).

Chapter 2 addresses the review of the most relevant ECC-based authentication schemes which are suitable for low-power devices. A modification is introduced to one of them to make it compatible with the hardware platform and ensure a lightweight computation. Also, a review of the Side Channel Attacks (SCA) that affects the said schemes is provided. Finally, an introduction to the principles of Residue Number System (RNS) is given. The RNS coprocessor used in this work is introduced herein as well.

Chapter 3 reviews some sort of modular inversion operations. A discussion on the security aspects of two of them is provided, and their vulnerabilities are analyzed. This chapter concentrates two of the main contributions of this thesis, due to the novelty of the Side Channel Attacks proposed to the target algorithms.

Chapter 4 addresses the ECC critical operations to be implemented as part of the authentication. Throughout this chapter a comprehensive explanation is given from the state-of-the-art algorithms to the final variations introduced. A contribution of the thesis is provided, through the reduction of an algorithm that allows a more efficient implementation of it and also makes it compatible with the HW platform. A countermeasure is applied to the final operation to ensure the mitigation of SCA.

Chapter 5 describes the HW platform used for the ECC operations evaluation. Performance results on the implementation of the operations are given, and a comparison with similar works is provided.

Finally, in the chapter 6 the Conclusions and Future work lines are given.

References

1. N. Komninos. Intelligent cities: Towards interactive and global innovation environments. International Journal of Innovation and Regional Development, 2009.
2. IoT for smart cities. https://www.nokia.com/networks/services/iot-for-smart-cities. Accessed in November 2020.
3. Smart cities. Sostenibilidad e innovación. https://www.i-de.es/redes-inteligentes/nuevos-modelos-energeticos/smart-cities. Accessed in November 2020.
4. B. Hamid, N. Jhanjhi, M. Humayun, A. Khan, A. Alsayat. Cyber Security Issues and Challenges for Smart Cities: A survey. 13th International Conference on Mathematics, Actuarial Science, Computer Science and Statistics (MACS), 2019.
5. The Open Web Application Security Project (OWASP). OWASP Top 10 Internet of Things - 2018. https://owasp.org/www-pdf-archive/OWASP-IoT-Top-10-2018-final.pdf. Accessed in November 2020.
6. https://www.bluetooth.com/learn-about-bluetooth/bluetooth-technology/bluetooth-security/blurtooth/. Accessed in November 2020.
7. C. Saatjohann, F. Ising, L. Krings, S. Schinzel. STALK: Security Analysis of Smartwatches for Kids. 15th International Conference on Availability, Reliability and Security (ARES 2020), Virtual Event, Ireland. ACM, New York, 2020.
8. J.-B. Bédrune, G. Campana. Everybody be Cool, this is a Robbery! https://www.sstic.org/media/SSTIC2019/SSTIC-actes/hsm/SSTIC2019-Article-hsm-campana_bedrune_neNSDyL.pdf. Accessed in November 2020.
9. A. Menezes, P. van Oorschot, and S. Vanstone. Handbook of Applied Cryptography. CRC Press, 1997.
10. A. Perrig, R. Szewczyk, V. Wen, D. Culler, J. D. Tygar. SPINS: Security protocols for sensor networks. Proceedings of ACM

Conference on Mobile Computing and Networks (MobiCom), pages 189–199, 2001.

11. R.L. Rivest, A. Shamir, and L. Adleman. A Method for Obtaining Digital Signatures and Public-Key Cryptosystems, 1977.
12. Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile. https://tools.ietf.org/rfc/rfc5280.txt, 2008.
13. A. Shamir. Identity-based cryptosystems and signature schemes. Proceedings of CRYPTO'84, Springer-Verlag, 1984.
14. D. Hankerson, A. J. Menezes, S. Vanstone. Guide to Elliptic Curve Cryptography, 1, Springer-Verlag New York, Inc. 2004.
15. N. Koblitz. Elliptic curve cryptosystems. Mathematics of Computation, 48, 203-209, 1987.
16. W. Diffie, M. Hellman. New directions in cryptography. IEEE Transactions on Information Theory, 22, 644–654, 1976.
17. A. D. Booth. A Signed Binary Multiplication Technique. The Quarterly Journal of Mechanics and Applied Mathematics, IV 2, 236–240, 1951.
18. L. Chen, D. Moody, A. Regenscheid, K. Randall. NIST Special Publication 800-186: Recommendation for Discrete Logarithm-based Cryptography: Elliptic Curve Domain Parameters. National Institute of Standards and Technology, Gaithersburg, Maryland), Draft, 2019.
19. E. Barker, D. Quynh. NIST Special Publication 800-57 (Part 3 Revision 1): Recommendation for Key Management: Application-Specific Key Management Guidance. National Institute of Standards and Technology, 2015.

2 Literature review

2.1 ECC-based authentication

2.1.1 ECDSA

ElGamal was the first authentication scheme proposed in 1984, which is based on the DLP [1]. The Digital Signature Algorithm (DSA), based on the DLP as well, was proposed in 1991 and later specified in the standard FIPS 186 [2]. These cryptosystems, along with Koblitz's discovering on ECC in [3], were the antecessors of the Elliptic Curve Digital Signature Algorithm (ECDSA), which was devised by Vanstone in 1992 [4].

ECDSA is probably the most widespread authentication method used in computer communications. Its efficiency is higher than exponentiation-based methods like RSA or ElGamal because it allows the use of shorter keys, as previously mentioned.

However, ECDSA does not solve by itself the problem of the public key validation. It is, a sender using ECDSA needs to publish its public key along with an authenticity certificate. Such scenario is not desirable for low-power devices. Nevertheless, due to compatibility reasons with some computer networks, many devices might still use the said authentication method [5-6]. In view of this fact, the ECDSA critical operations are considered in the chapter 4 of this thesis.

This method involves the modular inversion of a nonce, which is computed in the signature generation phase. The modular inversion is critical; the strength of the method strongly relies on the secrecy of the random value to be inverted. In the chapter 3 of this work, some security aspects related to the modular inversion are discussed.

2.1.2 Identity-based authentication

Certificate-less authentication methods are preferred for low power devices, due to the light weight in the transmission of the message signature. Shamir was the first to propose an identity-based scheme (IBS) for authentication [7]. In his method, instead of providing the public key along with its certificate, some identification data was used (name, email, etc.) to validate the key legitimacy. It is, the public key was built embedding that data already. The strength of Shamir's method relies in the impossibility of a third party to generate a valid public key with a fake ID, because a system private key is also needed in the generation process.

The IBS introduced in [8] also removes the need for a certificate, however, the method relies on Bloom filter [9] and Merkel tree [10] to preload each device with the public key information of the rest. This implies some constraints: the device needs extra memory and a secure mechanism to update the preloaded data every time the network scales with new nodes. Merkel hash tree needs a fixed number of nodes, while the Bloom filter length is computed considering the number of nodes; therefore, the method proposed in [8] is not suitable for dynamic wireless network in which the number of nodes may change.

The IBS method proposed in [11] seems to be a suitable one as it reduces the communication payload in the signature phase. This method is based on bilinear pairings, which is a costly operation and also vulnerable to some physical attacks as demonstrated in [12]. This IBS requires Boolean operations in the signature generation, and also requires to perform a point multiplication where the ECC point is a secret. The HW platform that is used in this work is efficient in some critical ECC operations, and provides an inherent protection against SCA; however, it cannot compute Boolean operations. On the other hand, the manipulation of a secret ECC point requires further countermeasures that reduce the performance of the system.

The first ECC-based IBS was proposed by Bellare in 2004 [13]. In the method called BNN-IBS, a sender identified by ID generates a signature (R, Y, z) to authenticate a message m as follows:

> Let P be a generator point of the curve $E(F_p)$, r a random in Z_p, x the system private key, $P_0 = xP$ the public system key, $R = rP$ the sender's public key, $c = hash_1(ID||R)$, and $s = r + cx$ the sender's private key. Then
>
> - Pick a random $y \in Z$ and computes $Y = yP$
> - Compute $z = y + hs$
> - where $h = hash_2(ID, m, R, Y)$.
> - The signature on message m is (R, Y, z).

In the verification phase, the receiver device does as follows:

- Computes $h = hash_2(ID, m, R, Y)$
- Computes $c = hash_1(ID||R)$
- Computes and verify whether $zP = Y + h(R + cP_0)$ holds

If it holds, the message is accepted, otherwise, it is rejected.

The proposal in [14] claims to be more efficient than BNN-IBS. The authors propose to build the signature like (R, h, z), reducing its size by trading the ECC point Y for the scalar h. In the verification phase is then proposed to compute and verify the equality $h = hash_2(ID, m, R, zP - hR - hcP_0)$.

In this thesis it is considered Bellare's method (BNN-IBS) rather than the one in [14]; however, a signature reduction is achieved as well. The variant proposed herein is to build the signature like (R, x_Y, z), where x_Y is the affine coordinate of Y. The procedure follows

- Pick a random $y \in Z$ and computes $Y = yP$
- Compute $z = y + hs$
- where $h = hash_2(ID, m, R, x_Y)$.

The signature on message m is (R, x_Y, z).

In the verification phase, the receiver device does as follows

- Computes $h = hash_2(ID, m, R, x_Y)$
- Computes $c = hash_1(ID||R)$
- Computes $Q = zP - hR - hcP_0$
- Verify whether $x_Y = x_Q$ holds

If it holds, the message is accepted, otherwise, it is rejected.

The operation to compute Q could be performed entirely in projective coordinates having at the output the point (X_Q, Y_Q, Z_Q). The use of projective coordinates is preferred as it avoids a division operation to compute the affine coordinates (see chapter 1). Thus, the equality verification $x_Y = x_Q$ could be performed like $x_Y \cdot Z_Q = X_Q$. Notice that the Y- coordinate is not needed, thus some computations can be saved in the multiplication.

2.2 Side Channel Attacks

Side Channel Attacks (SCA) are techniques oriented to retrieve sensitive information from cryptographic implementations in hardware devices. SCA take advantage of physical properties in hardware devices like power consumption, electromagnetic emanations or even the heat dissipation while a cryptographic operation is being performed [15-17]. The non-invasive measurements of these variables make this kind of attack a serious threat.

2.2.1 Side Channel Attacks

In a Simple Power Attack (SPA) the waveform of the power consumption (or electromagnetic radiation) is analyzed. The operations have a certain consumption pattern; it is, a multiplication of two integers takes longer time and higher power than the exclusive OR, for example. Also, if the target cryptographic primitive running is known, then its operations flow could be deduced and sensitive information could be extracted [18]. The Figure 2.1 shows an example of power trace corresponding to the Binary Extended Euclidean Algorithm (BEEA), where operations involving a secret are identified.

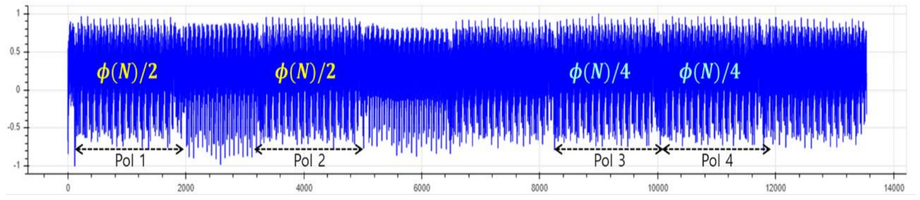


Fig. 2.1 Power trace of a Binary Extended Euclidean Algorithm implementation

2.2.2 Differential Power Analysis

The power consumption variates according to the data being manipulated even in the same operation. It is, when a register is loaded with 0xFF and its previous value was 0x00, the power related is not the same as if it was loaded with 0x11, for example. This different lead to a Differential Power Analysis, which is a statistical attack [15] [19].

2.2.3 Correlation Power Analysis

The Correlation Power Analysis introduced in [27] has the same basis as DPA, however it employs (typically) the Pearson correlation between the power measurements and a hypothesis of such consumption. CPA is proven to be more effective considering noisy signals than DPA. Figures 2.2 and 2.3 show examples of a CPA against a non-protected cypher and a protected one. The peak on Fig. 2.2 reveals the correlation among the assumption and the real value being computed, indicating with high probability the disclosure of the secret.

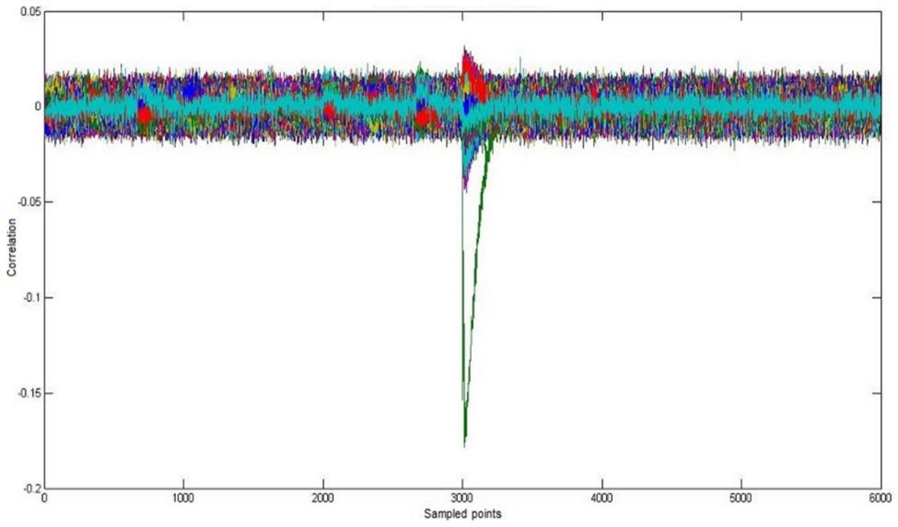


Fig. 2.2 Power trace of CPA against an unprotected AES implementation

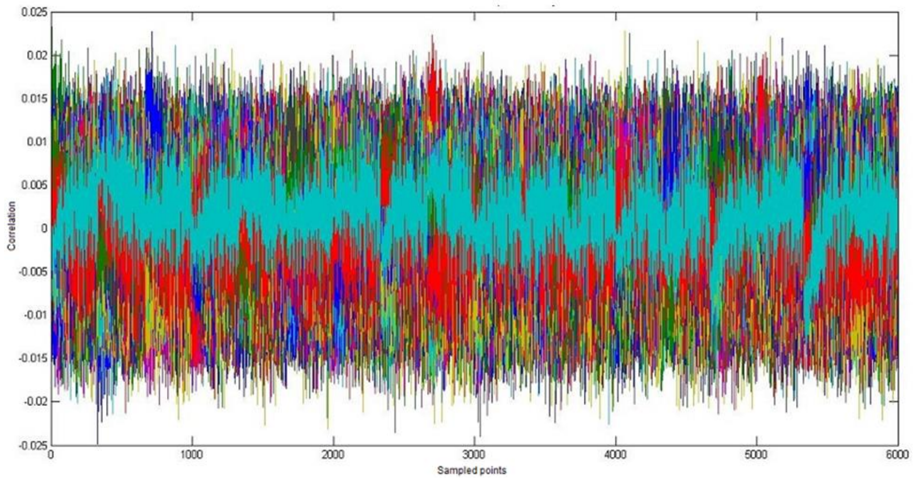


Fig. 2.3 Power trace of CPA against a protected AES implementation

2.2.4 Template Attack

In electronic devices, under certain conditions, a power consumption profile of some operations could be built in form of a template. With this, an adversary might be able to extract sensitive data from a target device by matching its power consumption with the computed templates. Such procedure is known as a Template Attack (TA). This kind of attack was introduced by Chari et al. in [20], and it is considered one of the strongest types of SCA so far. An ideal scenario to perform a TA is to acquire both profiling and attack power traces from the same device, in the same acquisition campaign. For this purpose, it is an advantage to have a profiling operation similar to the target one, which inputs and outputs could be controlled. Under such conditions the power consumption characterization would be close to optimal.

2.2.5 Side Channel Attacks vs ECC

A few of SCA have been published addressing the ECC [21]. Point multiplication is typically the target operation as it involves a secret, thus the attacks aim at recovering it. PM is especially vulnerable to a SPA if it is implemented using the straightforward doubling and adding algorithm, as it can be seen in Figure 2.4.

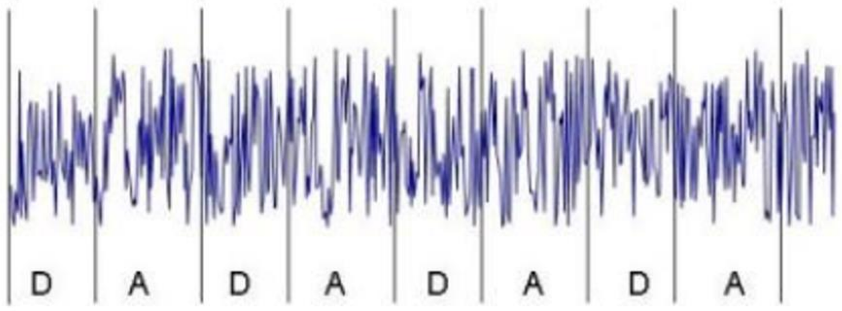


Fig. 2.4 SPA on an unprotected Point Multiplication

Some of the countermeasures to overcome SPA make indistinguishable the differences in Point Doublings (PD) and Point Additions (PADD) [22-23].

Also, DPA may target an ECC implementation as demonstrated in [24-25]. The DPA involved in these works can be counteracted through the randomization of the point coordinates, as determined by Coron in [26].

2.3 Residue Number System

2.3.1 Fundamentals

The Residue Number System is based on the Chinese Remainder Theorem and it allows the representation of non-negative integers through their remainders like

$$x_1 = X \, mod \, b_1$$
$$x_2 = X \, mod \, b_2$$
$$\dots$$
$$x_n = X \, mod \, b_n$$

where the base $B = \{b_1, b_2, \dots, b_n\}$ is a set of co-prime integers. Thus, $\{x_1, x_2, \dots, x_n\}$ is the representation of X in RNS. The dynamic range of B is defined as its Least Common Multiple

$$DR_B = \{b_1 \cdot b_2 \cdot b_3 \dots \cdot b_n\} \quad (2.1)$$

The basic operations (addition, subtraction and multiplication) between two integers X e Y represented in n-channels of RNS can be performed like

$$\{c_1, c_2, \dots, c_n\} = \{x_1 \circ y_1, x_2 \circ y_2, \dots, x_n \circ y_n\} \quad (2.2)$$

This implies that the computation can be parallelized, which implies an advantage in the operation speed [28]. The Figure 2.5 shows a high level block diagram of an RNS architecture.

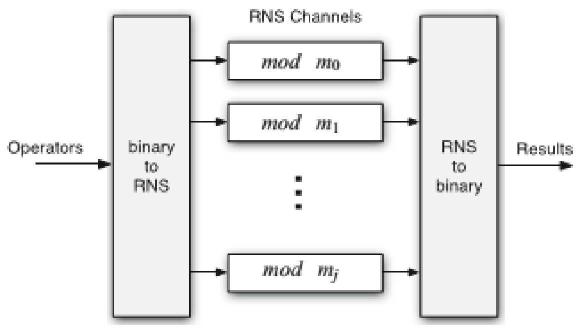


Fig. 2.5 RNS architecture

Before the computation, the integers have to be converted to RNS domain. Some values can be precomputed offline in this process, thus resulting a lightweight operation. In [29] more comprehensive details can be consulted.

The result of the RNS computation should be back converted to binary. This process is more complex than the previous one. Again, for more details, reader is forwarded to [29].

2.3.2 RNS resiliency against Side Channel Attacks

The importance of RNS for efficient cryptographic implementations have motivated the study of its resiliency against SCA. Researchers have demonstrated that the parallel circuits performing same operation make difficult an SCA [32-33]. Notice that, independent and isolated power consumption measurements of a single RNS channel are unfeasible to acquire. The rest of the channels perform the same operation with different operands, at the same time, thus acting as a significant source of noise. The common binary implementation on secure devices, need to implement a source noise, typically in the form of a random number generator.

Moreover, SPA and DPA attacks are proven to be defeated in RNS if the coordinates or the moduli set are randomized [34] [38-39]

2.3.3 The RNS coprocessor

RNS is a high-speed and efficient solution for large numbers multiplication, being more efficient than conventional two's complement arithmetic. The ability to obtain moduli sets with several channels results in high performance circuits, since each channel uses a reduced number of bits. The parallelism of operations reduces the latency of the system [30] [31].

In this work, we consider the RNS coprocessor described in [29] to evaluate the ECC PM and MPM operations intended for authentication schemes. The said coprocessor is a generic architecture, not exclusively designed for ECC implementations, thus another cryptosystems (e.g. RSA, ElGamal) could be implemented. Furthermore, the coprocessor can be configured to work with several ECC curve sizes up to 1024 bits, which perfectly fits the required standards for modern systems security. The flexibility of this solution makes it unique in the available literature [29].

The RNS coprocessor operates decoding a microcode sent from an external MPU. The microcode set the operation code and the operands. The data to be computed is also sent from the external MPU to the RNS coprocessor through a different bus.

Montgomery multiplication in RNS

The modular multiplication in the core of the RNS coprocessor is implemented following [35]. This is a RNS variant of the well-known

Montgomery multiplication [36]. Montgomery introduced a more efficient modular multiplication by trading $modulo\ N$ computations by $modulo\ L$, which allows more lightweight operations. In the RNS, L is the DR_B, thus the modular reductions for each element x_i are computed following

$$y_i = x_i\ mod\ b_i \quad (2.3)$$

Java emulator tool

An emulator and code generator for the coprocessor is presented in [37]. The tool offers a suitable interface to evaluate several cryptographic algorithms by coding them in Java. The relevance of this tool is that generates the microcode and formatted data to configure the coprocessor with the desired algorithm. The tool was used in this work to program the Point Multiplication and Multiple Point Multiplication algorithms evaluated. Annex A shows a picture of the interface.

Security considerations

As previously said, this is a general purpose arithmetic coprocessor, which is not specifically dedicated to cryptographic implementations, thus it lacks security countermeasures against SCA. Because the manipulated data is not masked in any sense, this should be considered to prevent a CPA or a potential Template Attack. Also, the moduli set is fixed in this version of the coprocessor, thus an alternative, based on (2.3) is the randomization of the coordinates, which is mandatory to avoid DPA.

Additionally, the generation of the microcode for PM might lead to a DPA like the one described in [24], which would apply in the external processor side. Although this threat is identified herein, the protection against SCA beyond the boundaries of the RNS processor is out of the scope of this thesis.

2.4 Summary

In this chapter, some ECC-based certificate-less authentication schemes were discussed. Bellare's BNN-IBS is a suitable one as it can be adapted to the RNS coprocessor without losing its efficiency. Moreover, this IBS can be implemented in the RNS applying the SCA countermeasures for a more complete protection.

A background of the state-of-the-art non-invasive physical attacks is also given herein. The main SCAs that apply to ECC-based cryptographic implementations, as quoted in some of the bibliography, can be counteracted by using the RNS method.

Finally, the fundamentals of the RNS is provided, and its advantages for the realization of cryptographic circuits are highlighted. A third-party RNS coprocessor is used to evaluate the critical operation involved in the authentication scheme.

The RNS coprocessor cannot compute divisions. This is an inconvenient for the implementation of the authentication scheme, specifically to provide the affine coordinates in the MPM operation (see chapter 4). In the case of ECDSA, the method requires a modular inversion, and it needs to be protected because a secret is manipulated. This issue motivated the investigation on the next chapter.

References

1. T. ElGamal. A Public Key Cryptosystem and a Signature Scheme Based on Discrete Logarithms. Advances in Cryptology. CRYPTO 1984. Lecture Notes in Computer Science, 196, Springer, Berlin, Heidelberg, 1984
2. Digital Signature Standard (DSS). Federal Information Processing Standards Publication 186 (FIPS 186), National Institute of Standards and Technology, 1994.
3. N. Koblitz. Elliptic curve cryptosystems. Mathematics of Computation, 48, 203-209, 1987.
4. S. Vanstone. Responses to NIST’s Proposal. Communications of the ACM, 35, July 1992.
5. P. T. Sharavanan, D. Sridharan, R. Kumar. A Privacy Preservation Secure Cross Layer Protocol Design for IoT Based Wireless Body Area Networks Using ECDSA Framework. J Med Syst, 42, 196, 2018.
6. E. Frimpong, A. Michalas. SeCon-NG: implementing a lightweight cryptographic library based on ECDH and ECDSA for the development of secure and privacy-preserving protocols in contiki-NG. Proceedings of the 35th Annual ACM Symposium on Applied Computing, New York, NY, USA, 2020.
7. A. Shamir. Identity-based cryptosystems and signature schemes. Proceedings of Advances in Cryptology (CRYPTO ’84), Springer-Verlag, 1984.
8. K. Ren, W. Lou, Y. Zhang. Multi-user broadcast authentication in wireless sensor networks. Proceedings of SECON’07, IEEE, 2007.
9. B. Bloom. Space/time tradeoffs in hash coding with allowable errors. Communications of ACM, 13, 422-426, 1970.
10. R.C. Merkle. A Certified Digital Signature. Proceedings of Advances in Cryptology (CRYPTO ’89), Lecture Notes in Computer Science, 435, 218-238, 1990.

11. S. Kyung-Ah, L. Young-Ran, P. Cheol-Min. EIBAS: An efficient identity-based broadcast authentication scheme in wireless sensor networks, Ad Hoc Networks, 11, 2013.

12. D. Jauvart, N. El Mrabet, J. J. A. Fournier, et al. Improving side-channel attacks against pairing-based cryptography. J Cryptogr Eng 10, 1–16, 2020.

13. M. Bellare, C. Namprempre, G. Neven. Security proofs for identity based identification and signature schemes. Proc. EUROCRYPT 2004, Springer-Verlag, 2004.

14. C. Xuefei, K. Weidong, D. Lanjun, Z. Bin. IMBAS: Identity-based multi-user broadcast authentication in wireless sensor networks, Computer Communications, Volume 31, Issue 4, 2008.

15. P. Kocher, J. Jaffe, B. Jun. Differential Power Analysis. Proceedings of Crypto 1999, 1666, 398–412, Santa-Barbara, CA, USA, 1999.

16. D. Agrawal, B. Archambeault, J. Rao, P. Rohatgi. The EM Side-Channel(s). Proceedings of CHES 2002, 2523, 29–45, Redwood City, CA, USA, 2002.

17. J. Knechtel, O. Sinanoglu. On mitigation of side-channel attacks in 3D ICs: Decorrelating thermal patterns from power and activity. 54th ACM/EDAC/IEEE Design Automation Conference (DAC), Austin, TX, 2017.

18. A. Cabrera Aldaya, R. Cuiman Márquez, A. Cabrera Sarmiento, S. Sánchez Solano. Side-channel analysis of the modular inversion step in the RSA key generation algorithm. I. J. Circuit Theory and Applications, 45, 2017.

19. T. Messerges, E. Dabbish, R. Sloan. Investigation of power analysis attacks on smartcards. Usenix Workshop on Smartcard Technology, 1999.

20. S. Chari, J. Rao, P. Rohatgi, Template Attacks. Cryptographic Hardware and Embedded Systems - CHES 2002, Lecture Notes in Computer Science 2523, Springer, Heidelberg, 2003.

21. L. Tawalbeh, H. Houssain, T. Al-Somani. Review of Side Channel Attacks and Countermeasures on ECC, RSA, and AES Cryptosystems. Journal of Internet Technology and Secured Transaction, 6, 2017.

22. C. Clavier, M. Joye. Universal exponentiation algorithm – A first step towards provable SPA-resistance. CHES 2001, 2162, 300–308, 2001.

23. D. Hankerson, A. J. Menezes, S. Vanstone. Guide to Elliptic Curve Cryptography, 1, Springer-Verlag New York, Inc. 2004.

24. K. Itoh, T. Izu, M. Takenaka. Address-Bit Differential Power Analysis of Cryptographic Schemes OK-ECDH and OK-ECDSA. CHES 2002, LNCS 2523. Springer, Berlin, Heidelberg, 2002.

25. J. Fan, B. Gierlichs, F. Vercauteren. To Infinity and Beyond: Combined Attack on ECC Using Points of Low Order. CHES 2011, LNCS 6917, Springer, Berlin, Heidelberg, 2011.

26. J. Coron. Resistance against differential power analysis for elliptic curve cryptosystems. CHES'99, LNCS 1717, 292–302, Springer, Berlin, Heidelberg, 1999.

27. E. Brier, C. Clavier, and F. Olivier. Correlation power analysis with a leakage model. CHES 2004, LNCS 3156, 16–29, Springer, Berlin, Heidelberg, 2004.

28. N. Szabo, R. Tanaka. Residue Arithmetic and Its Applications to computer Technology. McGraw-Hill. 1967.

29. P. Matutino, R. Chaves, L. Sousa. An Efficient Scalable RNS Architecture for Large Dynamic Ranges. Journal of Signal Processing Systems, 1–15, 2014.

30. A. A. Hiasat. VLSI implementation of new arithmetic residue to binary decoders. IEEE Transactions on Very Large Scale Integration Systems 13, 153–158, 2005.

31. A. Omondi, B. Premkumar. Residue Number Systems: Theory and Implementation. Imperial College Press, London, UK, 2007.

32. H. Pettenghi, J. A. Ambrose, R. Chaves, L. Sousa. Method for designing multi-channel RNS architectures to prevent power

analysis SCA. IEEE International Symposium on Circuits and Systems (ISCAS), Melbourne VIC, 2014.

33. J.C. Bajard, L. Imbert, P.Y. Liardet, and Y. Teglia. Leak resistant arithmetic. CHES 2004, LNCS 3156, 16–29, Springer, Berlin, Heidelberg, 2004.

34. J. Courtois, L. A. Abbas-Turki, J. C. Bajard: Evaluation of Resilience of randomized RNS implementation. IACR Cryptology ePrint Archive, 9, 2018.

35. K. Posch, R. Posch. Modulo reduction in residue number systems. IEEE Transactions on Parallel and Distributed Systems, 5, 449–454, 1995.

36. P. L. Montgomery. Modular Multiplication Without Trial Division. Mathematics of Computation, 170, 519–521, 1985.

37. J. Araujo, P. M. Matutino, R. Chaves. Residue Number System Hardware Emulator and Instructions Generator. 6th Conference on Trustworthy Manufacturing and Utilization of Secure Devices (TRUDEVICE 2016), 14-16, 2016.

38. G. Perin, L. Imbert, L. Torres and P. Maurine. Electromagnetic Analysis on RSA Algorithm Based on RNS. Euromicro Conference on Digital System Design, Los Alamitos, CA, 2013.

39. G. Perin, L. Imbert, P. Maurine, L. Torres. Vertical and horizontal correlation attacks on RNS-based exponentiations. Journal of Cryptographic Engineering, Springer, 5 (3), 171-185, 2015.

3 Modular inverse

The need to give a solution to the division problem in RNS, and to provide a secure mechanism for the nonce division in ECDSA, motivated the study of the modular inversion methods. A discussion about some representative examples of these algorithms is presented in this chapter, with focus on their efficiency and security. From the conducted study, and especially regarding the implementation security, two contributions were achieved.

In the first place, regarding the method described in section 3.3, a security analysis is provided which demonstrates potential vulnerabilities of a straightforward implementation. In addition, a contribution was done and formalized in a complementary work, introducing a secure variant for the inversion modulo 2^k[19].

On the other hand, the Euclidean algorithm is presented in subsection 3.4, where a security discussion about an RNS variant is conducted. An analysis considering the ECDSA inversion step as a mean to attack a coexistent RSA is also provided herein. Moreover, in another complementary work, it is introduced by the first time a Template Attack against a protected Euclidean algorithm, which targets the RSA key generation (See Annex B).

3.1 Introduction to modular inverse

In electronic devices that perform modular arithmetic, the division operation $c = a/b \; mod \; p$ is often solved through a modular inverse, like $c = a \cdot inv\_b \; mod \; p$. It is, instead of dividing by b, a modular multiplication is performed involving b's multiplicative inverse (inv_b), where $b \cdot inv\_b \; mod \; p = 1$ must hold.

Modular inverse is widely used in cryptography, and it is considered an expensive operation. The projective to affine coordinates conversion in ECC-based schemes require some modular inversions. Also, ECDSA makes the modular inversion of a nonce, which indeed is a sensitive value. Some multiplicative masking techniques also use this operation [1]. The Montgomery multiplication, which is extensively used for multi-precision operands, requires inversion as well [2].

In the remaining of this chapter some of the current state-of-the-art methods to compute the modular inverse are presented. The selected methods are considered among the most relevant ones for the sake of this work.

3.2 Fermat's Little Theorem (mod $prime$)

A well-known method to solve the modular inverse is the algorithm based on the Fermat's Little Theorem (FLT), although it works only for prime moduli [3]. The FLT states that, for any integer a and prime modulus p

$$a^p = a \;\; (mod \; p) \qquad (3.1)$$

And from there, it turns out that

$$a^{p-2} = a^{-1} \;\; (mod \; p) \, (3.2)$$

In many hardware devices (even in resource constrained ones), the modular inverse is performed as in 3.2, despite the exponential complexity of that method. Maybe the developers choose FLT when the modulus is a secret because the state of the art of SCA-protected exponentiations is well consolidated, as it is largely used in the RSA computation.

Recent bibliography shows that FLT is not the best option to perform modular inverse in residue arithmetic. In [4], a study was conducted to evaluate the FLT performance in an RNS-based hardware implementation. The comparison yielded that an RNS variant of FLT was about six times slower than a similar variant of the Euclidean algorithm.

3.3 Modular inverse (mod p^k). The ModInverse algorithm

A few methods exist in the bibliography to obtain the inverse modulo p^k [5], [6] and [7]. The particular case of the inversion modulo 2^k is quite useful to compute the Montgomery constant, which facilitates the modular multiplication through the Montgomery's method. The Montgomery inversion is another known method which has a variant (Almost Montgomery Inversion) that produces the modular inverse multiplied by 2^k [8]. The latter can be conveniently used to accelerate the computation. However, in terms of efficiency, the algorithm 6.11 given in [9] seems to be a better option to compute the inverse modulo 2^k. It seems that a minimal careful implementation should avoid SCA against this method. Its drawback is that it only works for modulo 2^k.

The ModInverse algorithm

The ModInverse algorithm in [7] was introduced by Koç in 2017. It is particularly interesting for its efficiency and because it works for modulo p^k, with any p and any k, which makes it more flexible than the previous similar methods.

It seems there is not an RNS variant proposed for this algorithm so far. Anyway, that topic is not in the scope of this work. For a future research on that line, it should be considered that a straightforward implementation of the original algorithm might be vulnerable to a secret retrieval. A security analysis addressing this issue is provided in the next subsection.

The assumptions to perform the computation by the ModInverse method are:

- p is a prime
- k is a positive integer
- $gcd(a;p) = 1 \quad (1 < a < p^k)$

The algorithm for modular inversions follows

Algorithm 3.1. ModInverse [mod p^k]

```
Input: $a, p$ and $k$; such that $gcd\ gcd\ (a;p)\ = 1$ and $a < p^k$
Output: $x = a^{-1}\ mod\ p^k$
1. $c = a^{-1}\ mod\ p$
2. $b_0 = 1$
3. $for\ i = 0\ \ to\ \ k - 1$
4.        $X_i = c \cdot b_i\ (mod\ p)$
5.        $b_{i+1} = (b_i - a \cdot X_i)/p$
6. $return\ x = (X_{k-1} \dots \dots \dots\ X_1\ X_0)_p$
```

The factor p is usually a small number (commonly 2 or 3), thus the computation at step 1 is expected to be easily performed. In fact, for the case of $p = 2$, the computation of c is trivial.

Special case $p = 2$

The previous algorithm can be reduced for $p = 2$. In that case, the inverse computation requires that $gcd(a; 2^k) = 1$, thus a must be an odd number and then $c = 1$. The simplified algorithm for $p = 2$ follows

<u>**Algorithm 3.2.** ModInverse [mod 2^k]</u>
Input: a and 2^k; such that $a < 2^k$ and a is odd
Output: $x = a^{-1}\ mod\ 2^k$

```
1. $b_0 = 1$
2. $for\ i = 0\ \ to\ \ k - 1$
3.        $X_i = b_i\ mod\ 2$
4.        $b_{i+1} = (b_i - a \cdot X_i)/2$
5. $return\ x = (X_{k-1} \dots \dots \dots X_1\ X_0)_2$
```

From the previous algorithm one appreciates that the operation at step 3 is trivial, as it only requires checking the LSB of b_i. On the other hand, the returned value in x is binary.

3.3.1 Security analysis for $p = 2$

In the following subsections we describe the two vulnerabilities found in the algorithm under analysis, that impede a safe manipulation of the sensitive data.

In a prospective RNS variant of ModInverse, those vulnerabilities could be exploited to disclose the random moduli set $M = (m_0, m_1, \dots \dots \dots\ m_n)$, thus making the implementation susceptible to SCA (see Chapter 3). Let us suppose the ModInverse is employed in the Montgomery multiplication, specifically to compute the Montgomery constants. Such constants are directly related to the RNS moduli set M, and they are computed like $m_i' = -m_i^{-1} \, mod \, 2^l$, where l is the channel`s bit length.

In the RSA-CRT scheme, those constants are directly related to the secret moduli p and q, and they are computed like $x' = -x^{-1} \, mod \, 2^k$, where k is the bit length of the respective moduli and x is either p or q. In [19] the author, describes this attack in an RSA-CRT scenario and gives a secure variant of the ModInverse algorithm to counteract SPA and Timing Attacks.

3.3.2 Asymmetric iterations

It is well known that an SPA allows to recover the secret k from the Square-and-Multiply method ($y = g^k$) due to a difference in the operations performed whether $k = 0$ or $k = 1$. The Montgomery ladder exponentiation solves that issue by always performing the same operations disregarding the value of k [10].

A similar issue has been detected in the inversion method under analysis in this work. It allows a straightforward SPA, which leads to an easy recovery of the operation result, and in consequence, the input data is disclosed. As from the previous section, the modular inverse of the input a, obtained through the algorithm 3.2 is formed by ______ $x = (X_{k-1} \dots \dots \dots X_1 X_0)_2$; where $X_i \in [0; 1]$. Furthermore, the intermediate result $b_i - a \cdot X_i$ at step 4 is always divisible by 2. At step 4, besides the multiplication $a \cdot X_i$, two other operations can be distinguished: a subtraction and a division by 2. The division can be performed as right shift because the result of the subtraction is always divisible by 2.

Regarding the subtraction, this may or may not be computed. One can see that if $X_i = 0$, then $a \cdot X_i = 0$, and then the subtraction $b_i - a \cdot X_i$ becomes $b_i - 0$. In a straightforward implementation of the original algorithm, the developer may choose to obviate the subtraction if $X_i = 0$. It is recalled that the original work does not refer to any SCA protection to keep the input data safe, thus it is believed the author did not consider a scenario with a secret input. If the subtraction is not performed, a significant difference in the execution flow exists depending on the X_i value. In summary

$$X_i = 0 \rightarrow b_{i+1} = (b_i - a \cdot 0)/2 = b_i/2 \quad \text{and}$$

$$X_i = 1 \rightarrow b_{i+1} = (b_i - a \cdot 1)/2 = b_i - a/2$$

Such a data-dependent characteristic could be distinguished in a power consumption trace of the algorithm execution. It would then lead to a straightforward SPA where the modular inverse of the secret could be directly recovered. Once the modular inverse is recovered, it is then trivial to obtain the input by computing $a = x^{-1} \, mod \, 2^k$. If a was a secret, as it is the case in the Montgomery constants computation (e.g. for RNS-based ECC or even the RSA-CRT), this would imply a critical security issue.

Nevertheless, the developer may choose a more regular implementation by always computing the subtraction. In this case, there are two possibilities: if $X_i = 0$, the subtraction $b_i - 0$ is computed, meanwhile, if $X_i = 1$, the operation computed is $b_i - a$. In the context of the Montgomery constants computation, from the second iteration, the operands b_i and a are large integers. This makes the subtraction $b_i - a$ highly susceptible of having lots of carry bits propagation. This effect has a negative impact on the latency of the additions/subtractions. While in $b_i - 0$ the carry propagation is null, in $b_i - a$ the carry propagation varies making that operation longer in time. This should be enough to apply a successful Timing Attack to distinguish one operation from the other, which directly lead to infer the values of the related X_i.

3.3.3 Operations latency

The latency of the arithmetic operations is closely related to the data length of the operands, especially in software implementations. In the case of additions/subtractions, they both commonly require managing a carry bit which is sequentially generated at each bit-bit operation. Therefore, the carry chain is as long as the operands, and it determines the whole operation latency.

Let us say, for example, that the evenness of an operand determines the next operation where it will be involved, and the said operation impacts on the operand's bit length. If that quantity is further added or subtracted from a constant value and this sequence is performed in a loop, the addition/subtraction latency might experiment variations at each iteration, as a consequence of the carry chain modification. If an adversary is able to identify the additions/subtractions through an SPA and measure those variations, then the operand's evenness (its Least Significant Bit - LSB -) might be traced back.

From the algorithm 3.2, one can see that the subtraction performed at the step 4 depends on b_i and a. The value of a is invariant throughout the whole operation, while b_i does varies. In fact, the value of b_i is strongly dependent on X_i. If $X_i = 0$, then b_{i+1}, computed at iteration i, yields $b_i = 2$. For consecutive values of $X_i = 0$, the respective b_{i+1} are always smaller by a factor of 2. On the other hand, considering $a < 0$ (as required in the Montgomery constant computation), it can be demonstrated that b_{i+1}, tends to a for consecutive $X_i = 1$.

Let's have $X_i = X_{i+1} = 1$. The correspondent calculations of b_i and b_{i+1} follow

$$b_i = \frac{b_{i-1}}{2} + \frac{a}{2} \qquad (3.3)$$

$$b_i = \frac{b_{i-1}}{4} + \frac{3a}{4} \qquad (3.4)$$

According to the right side of the equation 3.4 and comparing it with the right side of the equation 3.3, the subtrahend (which depends on a) in 3.4 is greater and it approaches more to a. The minuend is halved and tends to zero. Thus, it makes b_{i+1} closer to the value of a rather than to b_i. Something similar occurs when $X_i = 0$ and $X_{i+1} = 1$.

In summary, it might be expected to observe in a power trace, a continuous decreasing latency in the addition/subtraction for consecutive iterations where $X_i = 0$; while the latency would tend to increase for continuous $X_i = 1$ or even for transitions from $X_i = 0$ to $X_{i+1} = 1$.

The differences in the execution flow for $X_i = 0$ and $X_{i+1} = 1$ are enough to perform an SPA on algorithm 2. Thus, a timing analysis for this purpose is not necessary; however, in order to design a countermeasure to overcome such data-dependent vulnerability, the issue on the operations timing has to be taken into account.

3.4 The Euclidean algorithm

A binary variant of the Euclidean algorithm, introduced in [11], computes the greatest common divisor (GCD) of two integers. The extended variant (BEEA, Binary Extended Euclidean Algorithm) gives in addition, the inverse of one of the inputs modulo the other [12].

The Euclidean algorithm computes modular inversions by solving the Bézout's identity. This is:

$(u, v) = BEEA(a, m)$ where,

if $u \cdot a - v \cdot m = 1$, then

$u = \frac{1}{a}\ mod\ m$

The fact of working with prime and composite modulus, makes it quite versatile. The BEEA can be used on the ECDSA (to compute the inverse of the nonce k), which is maybe the most used ECC-based cryptosystem so far. Furthermore, the binary variant of the Euclidean algorithm is a suitable choice for hardware devices, as it substitutes the costly divisions by shifts allowing more efficient circuits. Moreover, it was recently demonstrated that a BEEA variant implementation performs six times faster than FLT in processors based on the residue number system [14].

In the remainder of this section, we will refer to a generic BEEA coded as in the Alg. 3.3, which is its classical definition, giving two outputs: the great common divisor of the inputs and the inverse of one of them [12].

Algorithm 3.3 Binary Extended Euclidean Algorithm

Inputs: k and p; where p is a prime

Outputs: $GCD(k, p)$ and $k_{inv} = k^{-1} \, mod \, p$

```
1.  $u = k$
2.  $v = p$
3.  $A = D = 1$
4.  $B = C = 0$
5.  $while \; u \neq 0 \; do$
6.      $while \; (u \; is \; even) do$ − − − − − − − − − − − − − u-loop
7.          $u = u/2$
8.          $if \; (A \; is \; even) \; and \; (B \; is \; even)$
9.              $A = A/2$
10.             $B = B/2$
11.         $else$
12.             $A = (A + p)/2$
13.             $B = (B - k)/2$ − − − − − − − − − − − − − end
14.     $while \; (v \; is \; even) \; do$ − − − − − − − − − − − − − − v-loop
15.         $v = v/2$
16.         $if \; (C \; is \; even) \; and \; (D \; is \; even)$
17.             $C = C/2$
18.             $D = D/2$
19.         $else$
20.             $C = (C + p)/2$
21.             $D = (D - k)/2$ − − − − − − − − − − − − − end
22.     $if \; u \geq v$
23.         $u = u - v$
24.         $A = A - C$
25.         $B = B - D$
26.     $else$
27.         $v = v - u$
28.         $C = C - A$
29.         $D = D - B$
30. $return \; (v = GCD(k, p), \; k_{inv} = C \; mod \; p)$
```

Notice that the oddness verification of the inputs is avoided for simplicity. The algorithm is written following a notation correspondent to the step of ECDSA where the nonce k is inverted modulo p.

When computing the inverse of k in ECDSA, like in the Alg. 3.3, the initial flow of the instructions code could be inferred. Let us consider $BEEA(k, p)$, where the ECC modulus is $p = \{p_{n-1} \dots p_1\, p_0\}$ and p_0 is the least significant byte. Because p is a prime, the v-loop will never be executed in the first iteration. However, in the first iteration, if k is even, the u-loop (steps 6 – 13) will be executed. Notice that $A = 1$ at the beginning, and that is enough condition to get into the ELSE branch of the u-loop, right after the shift. In that case, the operations at steps 12 and 13 are executed. The u-loop will execute until u gets odd. Once this condition is true, the flow will continue through step 22.

3.4.1 Security aspects of BEEA implementation

A large bit length difference in BEEA inputs has been exploited in [16, 18] to predict the execution flow of the algorithm and extract sensitive information. The countermeasures proposed so far, only focus on making both operands the same size, or they directly mask the sensitive input, to counteract such vulnerability.

The greatest common divisor is one of the outputs of the BEEA, thus, it is equivalent to write $GCD(a, b)$ instead $BEEA(a, b)$ to refer to such output from the Alg. 3.3. In [16] two methods were patented to protect the GCD from an SPA. The first one follows the property

$$GCD(X - 1, e) = GCD(X - 1 + r \cdot e, e) \qquad (3.5)$$

It consists on applying an additive masking to $X - 1$, where X is a secret prime (p or q) in an RSA scheme, e is the public key, and r is random. If r is a large nonce (e.g. $sizeof(X) - sizeof(e)$) then all the bits of the secret get masked. In such a case, even if an adversary is able to follow the complete algorithm flow, no relevant information will be disclosed.

Notice that, even assuming the case where an adversary obtains the masked $X' = X - 1 + r \cdot e$ through an SPA, and X' is reduced modulo e, still the brute force complexity to get X would make the attack unfeasible. This is true if considering the length of X has been properly chosen (e.g. ≥ 512 bits), and that $sizeof(X)$ is much larger than the $sizeof(e)$, which is the common scenario. The second method introduced by Chartier in [16] relies on the property

$$GCD(X - 1, e) = GCD(GCD(X - 1, r \cdot e), e) \qquad (3.6)$$

In [1] and [17] the proposed countermeasure to protect the BEEA from an SPA is based on the previous property as well. This countermeasure ensures that both operands have the same bit length to avoid the prediction of the execution flow. In this case, if the random number is a large nonce, then the specific conditional branch the algorithm takes at each iteration could not be guessed by an SPA on the power trace. This removes the vulnerability exploited in [17].

The methods in equations 3.5 and 3.6 are originally intended to protect the coprimality tests involving e and p and q candidates in RSA, also to protect the private key generation in the said cryptosystem. These countermeasures can also be used to protect the modular inversion of the nonce in ECDSA, given the case that it is performed through the Euclidean algorithm ($BEEA(k,p)$).

However, although the masking technique in eq. 3.6 prevents a SPA, it is highlighted that the secret is still manipulated in plaintext. This implies a high risk if profiling attacks are considered.

3.4.2 Profiling in ECDSA to attack the RSA

The manipulation of the public modulus p in plaintext in ECDSA, implies a risk for the RSA, given the case that both cryptosystems are implemented in the same device. As seen in Chapter 2, an operation which inputs and outputs can be controlled, is a good target to mount a Template Attack against a similar operation that manipulates a secret.

In ECDSA's modular inversion performed following Alg. 3.3 ($BEEA(k,p)$), when the nonce k is even, the operation at step 12 ($A = (1+p)/2$) is executed second, during the first iteration. Meanwhile, from the RSA key generation ($BEEA(e,\varphi(N))$), a suitable attack point for a TA could be the operation $C = (C + \varphi(N))/2$, where $C = 0$ in the first iteration. The high similarities among both operations allow using ECDSA to build a profile of its power consumption by choosing several values of p. The obtained profile can be later used to conduct a TA against an RSA key generation with the aim to disclose the secret $\varphi(N)$ while C is being computed.

The previous observation along with the non-protected secret issue in the masking method in 3.6 lead to formulate by the first time, a Template Attack against a BEEA implementation, which is described in Annex B.

Finally, it can be ensured that, to implement a secure inversion step in ECDSA and to avoid its use as a mean to attack a coexistent RSA, the two countermeasures proposed by Chartier in [16] must be used. Additionally, the multiplicative masking technique presented in [17] would also avoid the TA described in Annex B.

3.4.3 RNS variant of BEEA

The plus-minus algorithm introduced in [15] is a variant of the Euclidean method that avoids the large integer comparison (step 22 of Alg. 3.3). This is very useful for RNS-based systems, because the non-positional property of RNS makes difficult the comparisons. An extension of the original plus-minus algorithm, presented in [13], employs a modulo 4 verification to substitute the comparison between u and v. The plus-minus relies on the fact that two numbers x and y are odd whether $y + x$ or $y - x$ is divisible by 4.

Recently, the plus-minus idea let the authors of [14] to introduce an RNS variant of the Euclidean algorithm. The divisions by 2 and 4 were substituted by multiplications using their respective modular inverses (2^{-1} and 4^{-1}). On the other hand, the large number comparison was solved through a modulo 4 test. However, a modular reduction is also expensive in RNS, thus the authors reduced the computation by selecting all the elements m_i in the moduli set as odd and such that $|m_i|_4 = 1$.

The fact of imposing the $|m_i|_4 = 1$ restriction to all the elements m_i reduces the amount of candidates for the moduli set to one quarter.

This is contrary to the recommendation of having a random moduli set for security, as seen in chapter 3.

Regarding the efficiency, in [14] the authors verified that their BEEA-RNS variant runs 6 to 10 times faster than a FLT-RNS. However, in some cases, the FLT can be safely implemented using precomputed data to speed up the computation. One case could be the inverse of the nonce k in the ECDSA signature generation, or the inverse of the projective coordinate Z to calculate the affine x-coordinate in an ECC point multiplication.

3.5 Conclusions

<u>The ModInverse method</u>

The novel Modinverse algorithm presented in [7] could be a suitable choice for some systems, as it works for modulo p^k, with any p and any k, which is a flexible property. Its particular case (mod 2^k) makes it quite useful for the Montgomery multiplication. A prospective RNS variant of this method should not be discarded, as it would make this algorithm even more attractive. However, as discussed in this chapter, the method has some vulnerabilities that should be taken into consideration, as they impede a safe implementation to manipulate secrets. The work in [19] solves this threat and constitutes a side contribution of this thesis.

<u>Euclid’s and Fermat’s methods</u>

In this chapter, the fundamentals of the Euclidean algorithm have been introduced. The RNS variant of the BEEA presented in [14], demonstrated to be faster than an RNS-based FLT; however, the restrictions imposed to guarantee an efficient computation imply a potential risk, because the moduli set cannot be fully randomized. Therefore, a tradeoff solution should be found. An RNS-based FLT using precomputed data to speed up the calculations could be such an intermediate solution if the base, the exponent and the moduli set are randomized.

On the other hand, from the conducted analysis considering potential risks to other cryptosystems, it is suggested to apply further countermeasures in the BEEA implementation, to protect both the ECDSA and a coexistent RSA. The feasibility of a Template Attack to a partially protected BEEA implementation is demonstrated in Annex B. This contribution confirms the theoretical analysis regarding a BEEA implementation vulnerability.

References

1. D. Galindo, J. Großschädl, Z. Liu, P.K. Vadnala, S. Vivek. Implementation of a leakage resilient ElGamal key encapsulation mechanism. Journal of Cryptographic Engineering, 6(3), 229-238, 2016.
2. P. L. Montgomery. Modular multiplication without trial division. Math. Comput., 44, 519–521. 1985.
3. M. Liskov. Fermat's little theorem. Encyclopedia of Cryptography and Security. Springer, Boston, MA, 2005.
4. K. Bigou, A. Tisserand. Improving modular inversion in RNS using the plus-minus method. CHES 2013, LNCS 8086, Springer, Heidelberg, 2013.
5. S. R. Dussé, B. S. Kaliski Jr. A cryptographic library for the Motorola DSP56000. Advances in Cryptology – EUROCRYPT 90, Springer, 230-244, 1990.
6. O. Arazi, H. Qi. On calculating multiplicative inverses modulo 2^m. IEEE Transactions on Computers 57(10), 1435–1438, 2008.
7. Çetin Kaya Koç, A New Algorithm for Inversion mod p^k. IACR Cryptology ePrint Archive, 2017
8. B.S.Kaliski Jr., The Montgomery inverse and its applications. IEEE Transactions on Computers 44(8), 1064–1065; 1995.
9. T. St. Denis, G. Rose. Modular Reduction. BigNum Math: implementing cryptographic multiple precision arithmetic, Syngress Publishing, Rockland, USA, 2006.
10. P. L. Montgomery. Speeding the Pollard and elliptic curve methods of factorization. Math. Comput. 48, 243–264, 1987.
11. J. Stein. Computational problems associated with Racah algebra. J. Comput. Phys. 1(3), 397–405, 1967.
12. D. E. Knuth. The Art of Computer Programming, Seminumerical Algorithms, 2(3), Addison-Wesley Longman Publishing Co., Boston, MA, USA, 1997.

13. J. -P. Deschamps, G. Sutter. Hardware implementation of finite-field division. Acta Applicandae Mathematicae 93(1-3), 119–147, 2006.

14. B K. Bigou, A. Tisserand. Improving modular inversion in RNS using the plus-minus method. CHES 2013, LNCS 8086, Springer, Heidelberg, 2013.

15. R. P. Brent, H. T. Kung. Systolic VLSI arrays for polynomial GCD computation. IEEE Transactions on Computers, 33(8), 731-736, 1984.

16. M. Chartier. Method to protect a binary GCD computation against SPA attacks. Patent WO/2013/092265, Gemalto S.A. 2013.

17. A. Cabrera Aldaya, A. Cabrera Sarmiento, S. Sánchez-Solano. SPA vulnerabilities of the binary extended Euclidean algorithm. Journal of Cryptographic Engineering 7(4), 273–285, 2016.

18. A. Cabrera Aldaya, R. Cuiman Márquez, A. Cabrera Sarmiento, S. Sánchez Solano. Side-channel analysis of the modular inversion step in the RSA key generation algorithm. I. J. Circuit Theory and Applications, 45, 2017

19. S. de la Fé, C. Ferrer. A Secure algorithm for inversion modulo 2^k. Cryptography 2(3), 23, 2018.

4 ECC operations in the RNS coprocessor

In this chapter, the most critical elliptic curve operations involved in the ECC-based signature schemes are treated. The point multiplication (PM) and the multiple point multiplication (MPM) are considered critical operations as they both demand either a highly secure implementation or large computational resources.

An emphasis is made in the mechanisms adopted to guarantee the security against SCA, considering that these operations are computed by a general-purpose arithmetic coprocessor. There is also a focus on the algorithmic optimization of the ECC operations that lead to improve the signature generation/verification performance. The scope of this work is limited to the RNS coprocessor, therefore it is not considered herein the security of the external processor that feeds data and micro-codes to the RNS coprocessor.

4.1 Secure Point Multiplication

The last steps of the signature verification in ECDSA and BNN-IBS methods involve an ECC point comparison, which can be done by comparing only the x-coordinates (see chapter 2). Also, the signature generation in the said schemes, only needs for the x-coordinate of the curve generator point. This allows to implement the respective PM and MPM without computing the final y-coordinate. Therefore, in ECDSA and BNN-IBS, some modular multiplications can be saved. The expensive RNS-to-binary conversion of the non-needed y-coordinate can also be saved in both PM and MPM.

In the sequel, some references are made to M, S and I to denote the computational cost of field multiplications, squaring and inversions, respectively. Similarly, MM is used to denote the modular multiplications performed by the RNS coprocessor described in the chapter 3.

A more efficient PM performance can be obtained by using pre-computed data and techniques based on signed integer representation [8]. Those approaches need for more non-volatile memory space and further SCA countermeasures, to store, randomize and update the precomputed data. For some low power and memory constrained devices this might not be a suitable choice. Throughout this subsection we obviate such methods.

4.1.1 Random coordinates

One of the potential vulnerabilities identified in the RNS coprocessor (see chapter 3) is that it does not randomize the moduli set. This allows an adversary to perform a correlation attack against the PM to extract the bits of the secret scalar. The moduli set randomization acts as an input blinding technique; therefore, we directly randomize the generator point coordinates to achieve the same effect and counteract such correlation attacks.

Based on the ECC points property of having n representatives in n Z-planes, the generator point is randomized before manipulating the scalar bits. Nevertheless, depending on the security level required, the intermediate results might also be randomized multiple times during the PM with a negligible cost. The new Jacobian projective coordinates are computed as follows

- *Generate a random number* rnd
- $X' = rnd^2 \cdot X(mod\ N)$
- $Y' = rnd^3 \cdot Y(mod\ N)$
- $Z' = rnd \cdot Z(mod\ N)$

4.1.2 Montgomery ladder with (X, Y)-only variant

The co-Z addition

Meloni introduced in [1] a reduced addition technique for different points sharing the same Z-coordinate, which can be described as follows:

Let $P = (X_1, Y_1, Z)$ and $Q = (X_2, Y_2, Z)$ be two points on the same curve in the same Z-plane. The addition $P + Q = (X_3, Y_3, Z_3)$ can be computed through

$$X_3 = (Y_2 - Y_1)^2 - (X_2 - X_1)^3 - 2X_1(X_2 - X_1)^2$$
$$Y_3 = (Y_2 - Y_1)(X_1(X_2 - X_1)^2 - X_3) - Y_1(X_2 - X_1)^3$$
$$Z_3 = Z(X_2 - X_1) \qquad (4.1)$$

Notice that, as part of the point addition, a quantity $Z(X_2 - X_1)$ is obtained, and also the intermediate values $X_1(X_2 - X_1)^2$ and $Y_1(X_2 - X_1)^3$. This clearly reveals a representative (P') of the point P with the same Z-coordinate of $(P + Q)$, where the new $P' = (\lambda^2 X_1, \lambda^3 Y_1, \lambda Z)$, with $\lambda = (X_2 - X_1)$. This, in fact, was the key observation of Meloni, as it allows to perform chained ZADD (co-Z addition) to solve a point multiplication more efficiently.

ZADD's cost is 5M + 2S and it requires only 6 field registers. Because the RNS coprocessor performs only modular multiplications, 7MM are counted to complete the ZADD.

Hereinafter, reference is made to ZADD instead of ZADDU to be aligned with the notation in [2], because a few references are made to that work. The same happens with the rest of the algorithms from [2] mentioned below.

The conjugate addition

Goundar et al. introduced the co-Z conjugate addition (ZADDC) in [2]. The operation yields the points $R = Q + P$ and $S = Q - P$, from two points $P = (X_1, Y_1, Z)$ and $Q = (X_2, Y_2, Z)$ which share the same Z-coordinate. The cost of ZADDC is 6M + 3S (9MM) and it requires 7 field registers.

Moreover, the resultant points R and S do share the same Z-coordinate as well. From these results, it is easy to see that a further addition $R + S = 2Q$, along with the intermediate result $Q + P$, correspond both to the per bit operations to perform a point multiplication ($Q = k \cdot P$) through a well-known method: Montgomery ladder.

Co-Z Montgomery ladder

The mentioned authors realized that the Montgomery ladder main loop could be written in terms of the ZADDC and the ZADDU operations. Considering the points P and Q are manipulated from the point registers R_b and R_{1-b}, where the sub-index b denotes the current bit value of the scalar, and T is a temporary register, the main loop evaluates

$(R_{1-b}; T) \leftarrow ZADDC(R_b; R_{1-b})$, followed by

$(R_b; R_{1-b}) \leftarrow ZADDU(R_{1-b}; T)$

The total cost of the operations into the Montgomery ladder loop is n(11M + 5S), which are n(16MM) in the RNS coprocessor, where $n = sizeof(k) - 1$ expressed in bits.

In [3] the authors had noticed that the Z-coordinate was not used in ZADDU and ZADDC. That observation motivated a more efficient variant of the co-Z Montgomery ladder. The new variant is based on the combination of ZADDU and ZADDC algorithms but removing the computation of the Z-coordinate. Such coordinate is only recovered at the end, out of the main loop. This enhances the PM performance by reducing two modular multiplications at every loop iteration in the Montgomery ladder. The respective operations without handling the third coordinate are denoted by ZADDU' and ZADDC'. This contribution is remarkable, even despite the further calculations needed to recover the Z-coordinate.

In a step beyond in [2], the new ZADDU' and ZADDC' were combined following

$$\left(R_{k_0}; R_{1-k_0}\right) \leftarrow ZADDC'(ZADDU'(R_{k_0}; R_{1-k_0})) \qquad (4.2)$$

to obtain a single operation: ZACAU', which trades 1M by 1S and saves one field register as its main advantages. ZACAU' performs 14MM in the main loop.

In this new variant of Montgomery ladder using ZACCAU', the scalar's LSB is separately (out of the loop) processed to recover the Z-coordinate.

The same work also proposes a slightly more efficient algorithm (ZDAU') which saves 10MM from the overall computation in respect to ZACAU'. ZDAU' follows the same principle of doubling-adding by mean of ZADDU' and ZADDC'. The key difference is that ZDAU' makes use of signed scalar. This method requires a sign inversion $(R_1 \leftarrow (-R_1))$ for certain combinations of the scalar bits. Such irregularity should be carefully managed, as it can be distinguished and exploited in a SCA.

4.1.3 Co-Z Montgomery ladder in the RNS coprocessor

To trade a field multiplication by a field square as it is done in ZACAU' variant, does not represent any advantage in the RNS coprocessor, because it performs both operations with the same computational cost, as mentioned in the chapter 3. Besides, ZACAU' occupies more memory to store the operation microcode for the coprocessor. It is, the Montgomery ladder main loop, considering a variant like ZADDC'/ZADDU', requires 24+13 microcode operations, while for the ZACAU' variant, another 46 should be added. But what it is more important, the ZACAU' algorithm, once the register allocation is done (see line 44 of Alg. 26 in [2]), has an operation that exceeds the bounds of the dynamic range in the RNS coprocessor. This fact impedes the use of the algorithm in the RNS coprocessor.

On the other hand, the ZDAU' algorithm might not represent a significant improvement to justify the potential vulnerability of the irregular sign inversion it requires. ZDAU’ outperforms ZACAU’ by only 10MM. A PM involving a 256-bits scalar, which is a common scalar size, needs an amount of 3584MM in the main loop of both algorithms. This figure makes a 10MM difference almost negligible. Besides, the RNS coprocessor cannot perform conditional instructions, therefore the point inversion should be explicitly coded. This fact would be easily recognized in the microcode and it could leak information about the scalar bits being computed.

To avoid the previous issues, this work adopts the approach based on the Montgomery ladder (Alg. 15 in [2]). However, the pair ZADDC’/ZADDU’ with the X, Y-only variant is used, instead of ZACCAU’. The Z-coordinate is recovered once the main loop is completed. By this way, we can access to the outputs of the ZADDC’ algorithm when k_0 is being processed (out of the loop), to effectively recover the third coordinate. Moreover, the ECC generator point is randomized, following section 4.1.1, previous to the manipulation of the scalar bits. The reduced PM follows the Algorithm 4.1.

Because in the signature schemes in the scope of this work, the affine y-coordinate is not used, the Alg. 4.1 can save some operations. In ZADDU’ the last five field operations are dedicated to compute the projective Y-coordinates of the output points (see Alg. 19 in [2]). These operations are not necessary when the point addition procedure is invoked in the step 11 of the Alg. 4.1. The reduced algorithm for the last point addition is denoted by rZADDU’. A total of 5 operations, including 2MM are reduced. Furthermore, there is no need to convert from RNS to binary the resultant y-coordinate, as it is not computed.

Algorithm 4.1 Reduced Co-Z Montgomery ladder

Input: $P = (x_P; y_P) \in E(F_q)$; $rnd \in N$; *and* $(k_{n-1;} \dots k_{1;} k_0)$ such that $k_{n-1} = 1$

Output: $Q = k \cdot P$

```
1.  $(R_1, R_0) \leftarrow DBLU'(P)$
2.  $(R_1, R_0) \leftarrow Rand(R_1, R_0, rnd)$
3.  $for\ i = n-2\ downto$ 1 do
4.      $b \leftarrow k_i$
5.      $(R_{1-b}, R_b) \leftarrow ZADDC'(R_b, R_{1-b})$
6.      $(R_b, R_{1-b}) \leftarrow ZADDU'(R_{1-b}, R_b)$
7.  $end\ for$
8.  $b \leftarrow k_0$
9.  $(R_{1-b}, R_b) \leftarrow ZADDC'(R_b, R_{1-b})$
10. $Z = x_P Y(R_b)\big(X(R_0) - X(R_1)\big)$
11. $\lambda = y_P X(R_b)$
12. $(R_b, R_{1-b}) \leftarrow rZADDU'(R_{1-b}, R_b)$
13. $return\ x_Q = \left(\frac{Z}{\lambda}\right)^2 X(R_0)$
```

This approach outperforms the results in [4] and [5], which propose both a main loop with 20MM per bit of the scalar. Although in these works the PM is performed by mean of the x-only Montgomery ladder variant, and our proposal adds the y-coordinate too, the overhead of the binary-to-RNS conversion assumed by our approach, is negligible when the main loops in both variants are compared [6]. See the Table 4.1 for a detailed comparison.

Table 4.1. Comparison of SCA-protected point multiplication algorithms

Regular PM methods	# regs.	Total cost in RNS
[2] (X, Y)-only co-Z Montgomery ladder (ZACAU')	6	n(14MM) + 8MM + 1I
[4] x-only Montgomery ladder in RNS[1]	13	n(20MM) + 1I
[5] x-only Montgomery ladder in RNS[1]	n/a	n(20MM) + 1I
This work (X, Y)-only reduced co-Z Montgomery ladder (ZADDU'/ZADDC')	6	n(14MM) + 4MM + 1I

[1] *Considering the cost of computing only the affine x-coordinate*
[2] *n: number of scalar bits*

The following steps summarize the procedure to perform a point multiplication, following our reduced co-Z Montgomery ladder, in the RNS coprocessor

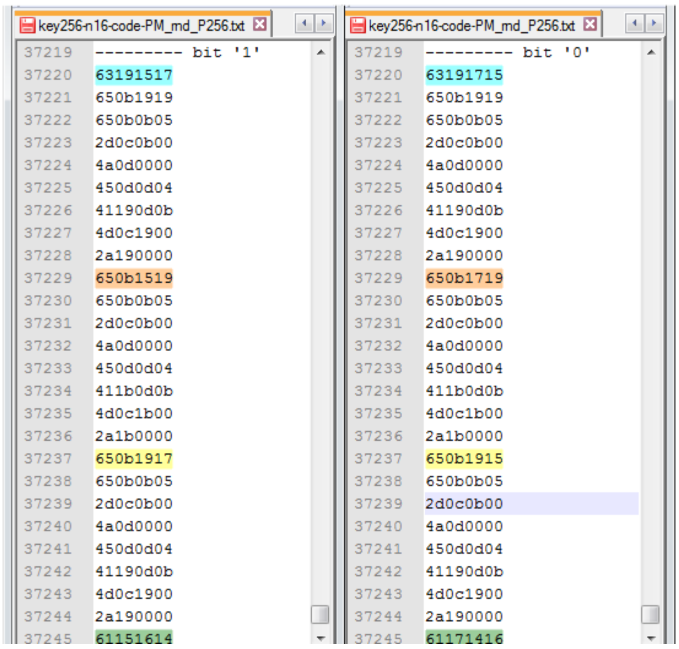
- Convert ECC parameters (a, b) and x- and y-coordinates from binary to RNS
- Represent a, b and x- and y-coordinates in the Montgomery domain
- Perform the PM following the Alg. 4.1
- Convert x_Q from Montgomery domain to the original domain
- Convert x_Q from RNS to binary

4.1.4 Masked registers against SCA in Point Multiplication

In the Alg. 4.1, into the loop, the manipulation of the registers R_0 and R_1 by ZADDC' and ZADDU' depend directly on the current bit of the secret scalar that is being evaluated. Such dependency could be easily exploited by a SCA.

The straightforward variant to perform a PM in the RNS coprocessor is to use two separate microcode sets: one for $k = 0$ and the other for $k = 1$, as it can be seen in the Figure 4.1. From the figure, the correspondence among the scalar bits and the registers denoted by 0x15 and 0x17 can be easily noted.

One effective technique to counteract this threat is to apply a random mask to the sensitive data. A masking countermeasure is strongly recommended to protect the microcode. Notice that the parallelism of the RNS channels operation, which give resilience against the SCA, does not protect the microcode load onto the RNS processor.



key256-n16-code-PM_md_P256.txt
key256-n16-code-PM_md_P256.txt

```
--------- bit '1'
63191517
650b1919
650b0b05
2d0c0b00
4a0d0000
450d0d04
41190d0b
4d0c1900
2a190000
650b1519
650b0b05
2d0c0b00
4a0d0000
450d0d04
411b0d0b
4d0c1b00
2a1b0000
650b1917
650b0b05
2d0c0b00
4a0d0000
450d0d04
41190d0b
4d0c1900
2a190000
61151614
```

```
--------- bit '0'
63191715
650b1919
650b0b05
2d0c0b00
4a0d0000
450d0d04
41190d0b
4d0c1900
2a190000
650b1719
650b0b05
2d0c0b00
4a0d0000
450d0d04
411b0d0b
4d0c1b00
2a1b0000
650b1915
650b0b05
2d0c0b00
4a0d0000
450d0d04
41190d0b
4d0c1900
2a190000
61171416
```

Figure 4.1. Section of two microcode sets ($k = 1$ and $k = 0$) for a ZADDC' implementation in the RNS coprocessor

4.2 Multiple Point Multiplication

Multiple Point Multiplication is the main operation in the verification phase of ECC-based signature schemes. As there is no secret disclosure risk when performing MPM, some techniques apply for a faster computation, like Shamir's trick and Interleaving [7-8]. Such techniques are mainly based in the use of precomputed values instead of doing straightforward calculation.

Precomputation-based techniques for MPM are more efficient than a double PM but still require a heavy computation. In this section the implementation of MPM in the RNS coprocessor is discussed.

4.2.1 Interleaving with NAF

Interleaving is a technique to speed up both PM and MPM, through the use of precomputed values. The combination of Interleaving with Non Adjacent Form (NAF) improves the performance of the computation. NAF allows the representation of scalars in a way that maximizes the number of zeroes, while using negative elements. In [8] a comprehensive explanation of Interleaving and NAF is provided.

In the Table 3.6 in [8], Interleaving using NAF arises as one of the more efficient methods in terms of computational speed and storage to perform a MPM.

From the Alg. 4.2, in line 9, it is easy to see that the whole loop is omitted when the evaluated bits of the scalars are all zeroes. The advantage of the NAF relies just there, in placing as much zeroes as possible, to reduce the computational effort.

Algorithm 4.2 Interleaving with NAF

Input: v, k^j, w_j and ECC points P_j; such that $1 \leq j \leq v$

Output: $\sum_{j=1}^{v} k^j P_j$

```
1.  Compute $iP_j \ for \ i \in \{1, 3, \ldots, 2^{w_j - 1} - 1\}$
2.  Compute $NAF_{w_j}(k^j) = \sum_{i=0}^{l_j - 1} k_j^j 2^i$
3.  $Let \ l = max\{l_j : 1 \leq j \leq v\}$
4.  $Define \ k_i^j = 0 \ for \ l_j \leq i < l, 1 \leq j \leq v$
5.  $Q \leftarrow \infty$
6.  $for \ i = l - 1 \ \ downto \ 0 \ do$
7.      $Q \leftarrow 2Q$
8.      $for \ j = 1 \ \ to \ v \ do$
9.        $if \ k_i^j \neq 0 \ then$
10.         $if \ k_i^j > 0 \ then \ Q \leftarrow Q + k_i^j P_j$
11.         $else \ Q \leftarrow Q - k_i^j P_j$
12.     $return \ Q$
```

4.2.2 Composite operations for Interleaving

Longa and Miri improved the performance of some ECC operations in [9], including the reduction of the successive Point Doublings (PD) to 3M plus 5S (8MM in the RNS coprocessor). On the other hand, the authors demonstrated in [10] the feasibility of fixing $a = -3$ in some ECC curves, with which the PD can be reduced as in [9]. The cost of the standard PD in Jacobian coordinates is 10MM.

Longa and Miri also developed a method to improve the computation of $dP + Q$ which is a recurrent and critical operation for PM and MPM [11]. The researchers were based on Meloni's finding about the co-Z Point Addition (ZADD), which is more efficient than the general addition [1]. Their key observation was to recognize an equivalent representation of P from the computation of $P + Q$ at no extra cost.

Let $P = (X_1, Y_1, Z_1)$ and $Q = (X_2, Y_2)$ be two points on the same ECC curve. The operation $P + Q = (X_3, Y_3, Z_3)$ is performed as a mixed Jacobian – affine following

$$X_3 = 4(Z_1^3 Y_2 - Y_1)^2 - 4(Z_1^2 X_2 - X_1)^3 - 8X_1(Z_1^2 X_2 - X_1)^2$$
$$Y_3 = 2(Z_1^3 Y_2 - Y_1)(4X_1(Z_1^2 X_2 - X_1)^2 - X_3) - 8Y_1(Z_1^2 X_2 - X_1)^3$$
$$Z_3 = 2Z_1(Z_1^2 X_2 - X_1) \qquad (4.3)$$

$P' = (X', Y', Z') = 4X_1(Z_1^2 X_2 - X_1)^2, 8Y_1(Z_1^2 X_2 - X_1)^3, 2Z_1(Z_1^2 X_2 - X_1)$ is obtained from the intermediate calculations in (4.3) and it is a representative of the point P. Recall that an ECC point $P' = (X', Y', Z')$ is a representative of $P = (X, Y, Z)$ if $(X', Y', Z') = (\lambda^2 X, \lambda^3 Y, \lambda Z)$. Notice that $\lambda = 2\left(Z_1^2 X_2 - X_1\right)$ in P'.

Based on this, Longa and Miri proposed the computation $dP + Q = P + P \ldots + (P + Q)$, where d is a small integer. $(P + Q)$ is computed first as a Mixed-Affine Point Addition (MA-PADD), and the result is computed backwards using ZADDs with every (updated) P.

In a step beyond, the researchers defined a joint operation for the case $2P + Q$ [11]. The cost of the standard operation would be 1PD + 1MA-PADD = 10MM + 11MM. The improved operation is computed with only 18MM through an Unified Addition (UADD) of $P + P + Q$.

Note that an implementation of the Alg. 4.2 may include $2P + Q$ in the lines 10 and 11, therefore, UADD leads to a significant reduction of the computation in the MPM.

4.2.3 MPM implementation in the RNS coprocessor

The MPM was evaluated in the RNS coprocessor for the signature verification stage of ECDSA and BNN-IBS. In both implementations, the improvements on $2P + Q$ proposed in [11] were used. The scalars were transformed through NAF according to the coprocessor's resources available.

ECDSA

The MPM in ECDSA is like $uP + vQ$, thus it requires precomputed values derived from two ECC points in affine coordinates. A 4-NAF transformation was applied to the scalars, therefore the points $P, 3P, 5P, 7P, Q, 3Q, 5Q, 7Q$ and their respective negatives in affine coordinates were used, occupying all the 64 registers of the coprocessor. On the other hand, the strategy to compute the combined multiplication was implemented following

$$if\ \ (u_i \neq 0\ \ and\ \ v_i = 0)\ \ then\ R \leftarrow UADD(R, u_iP)$$

$$if\ \ (u_i = 0\ \ and\ \ v_i \neq 0)\ \ then\ \ R \leftarrow UADD(R, v_iQ)$$

$$if\ \ (u_i \neq 0\ \ and\ \ v_i \neq 0)\ \ then\ \ R \leftarrow MA\_PADD(UADD(R, u_iP), v_iQ)$$

Note that when $u_i \neq 0$ and $v_i \neq 0$, the standard computation of _ $2R + u_iP + v_iQ$ would cost 1PD + 2MA-PADD (10MM + 22MM), while the use of UADD (1UADD + 1MA-PADD -> 28MM) saves 4MM.

BNN-IBS

The largest MPM computation in BNN-IBS is like $z_iP - h_iR - l_iP_0$, therefore, an Interleaving with NAF required more registers for precomputed values than ECDSA. A 3-NAF transformation was the maximum that could be applied to the scalars, thus allowing to use the precomputed $P, 3P, R, 3R, P_0, 3P_0$ and their respective negatives, at the cost of 60 out of the 64 processor registers. The computation of the combined multiplication followed the same strategy than for ECDSA when one or two scalars were different from zero. For the triple multiplication, the strategy followed

```
$if\ (z_i \neq 0\ and\ h_i \neq 0\ and\ l_i \neq 0)\ \ then$
        $Y \leftarrow MA\_PADD(MA\_PADD(UADD(Y, z_iP), h_iR), l_iP_0)$
```

In this case, the standard computation of $2Y + z_iP - h_iR - l_iP_0$ would cost 1MA-PADD (11MM) more than in ECDSA. The use of UADD (1UADD + 2MA-PADD -> 39MM) again saves 4MM to this operation.

4.3 Conclusions

In this chapter, an implementation of the PM in the RNS coprocessor was analyzed from a security and performance point of view. The need for random coordinates was discussed to avoid correlation attacks. Moreover, following a set of efficient state-of-the-art methods for intermediate computations, a reduced and secure co-Z Montgomery ladder algorithm was proposed. This algorithm saves 4MM in the point multiplication and avoids the computation of the *y*-coordinate, which also implies the avoidance of the expensive transformations from the Montgomery domain. In addition, a potential vulnerability was identified in the load of microcode into the processor, which might lead to a correlation attack to infer the secret scalar's bits. A masking solution was proposed to mitigate such threat.

The MPM was also analyzed herein, and some improvement was achieved in terms of its performance. The efficient method discussed to compute $2P + Q$ was extended to fit the implementation of Interleaving with NAF for ECDSA and BNN-IBS. Also, the NAF transformation and precomputed values were calculated to use as much of the processor's resources as possible. The significant result is that MPM can be performed faster during the verification stage of the said signature schemes.

References

1. N. Meloni. Fast and Secure Elliptic Curve Scalar Multiplication over Prime Fields using Special Addition Chains. Cryptology ePrint Archive, Report 2006/216, 2006.
2. R. R. Goundar, M. Joye, A. Miyaji, M. Rivain, A. Venelli. A scalar multiplication on Weierstraß elliptic curves from co-Z arithmetic. J. Cryptogr. Eng. 1(2), 161–176, 2011.
3. A. Venelli, F. Dassance. Faster side-channel resistant elliptic curve scalar multiplication. Contemporary Mathematics 521, 29-40, 2010.
4. S. F. Antão. High-performance and Embedded Systems for Cryptography. Ph.D. dissertation, Lisbon Tech (2013)
5. J. C. Bajard, S. Duquesne, M. Ercegovac. Combining leak-resistant arithmetic for elliptic curves defined over Fp and RNS representation. Publications Mathématiques de Besançon: Algébre et Théorie des Nombres, 67-87, 2013.
6. P. Matutino. Residue Number Systems. Efficient Architectures and Circuits. Ph.D. dissertation, Lisbon Technical Institute, 2015.
7. A. Shamir. Shamir’s Trick. Encyclopedia of Cryptography and Security. Springer, Boston, MA, 2011
8. D. Hankerson, A. J. Menezes, S. Vanstone. Guide to Elliptic Curve Cryptography, 1, Springer-Verlag New York, Inc. 2004.
9. P. Longa, A. Miri. Fast and Flexible Elliptic Curve Point Arithmetic over Prime Fields. IEEE Transactions on Computers, 57(3), 289-302, 2007.
10. O. Billet, M. Joye. Fast Point Multiplication on Elliptic Curves through Isogenies. AAECC 2003, LNCS 2643, 43–50, Springer, Heidelberg, 2003.
11. P. Longa, A. Miri. New Composite Operations and Precomputation Scheme for Elliptic Curve Cryptosystems over Prime Fields. Proceedings of The 11th International Workshop on Practice and Theory in Public-Key Cryptography, PKC 2008.

5 HW evaluation platform

In this chapter the hardware setup for the ECC operations evaluations is presented. Also, the performance on the PM for the standard NIST curve P-256 is given. A discussion about the impact of the technology in the operations result is also provided. The reader should be familiarized with the FPGA technology used in this chapter. FPGA devices are widely employed since many years ago, due to the feasibility to conduct the evaluation of a hardware system without the need to build an expensive ASIC.

5.1 FPGA-based hardware platform for ECC evaluation

The board Xynergy-M4, from Silica used for the ECC operations evaluation is built around a Xilinx Spartan-6 FPGA (xc6slx75) and a microprocessor unit (MPU) ARM Cortex-M4 [7]. Among the available resources when the experimental was started, the Xynergy-M4 offered a good solution for the implementation of the RNS coprocessor. Recalls that it needs an external processor to send the microcode and data to operate.

The system frequency was configured at 88,3 MHz being the maximum accepted. Xilinx ISE tools were used to build and simulate the project and configure the device.

The RNS coprocessor was embedded in a Xilinx ISE project through its VHDL code definition. An interface IP module was developed in VHDL for the manipulation of the data and microcode, and to provide control signals to the external MPU. The interface module receives the microcode and data from the MPU through SPI. The interface module also implements two blocks that handle FIFO signals to interact with the RNS coprocessor.

In the case of a PM, the MPU generates a random number for the scalar. For the MPM, the MPU acting as a node, is expected to receive the elements of a digital signature including the scalars and the ECC points.

The MPU configures a DMA module to send the data and microcode to the coprocessor depending on the operation to be performed. The microcode sets for each operation are previously loaded into a RAM. When the RNS output is ready, the interface sends an interrupt to the MPU, which receives the ECC points coordinates. The Figure 5.1 shows a block diagram of the hardware setup.

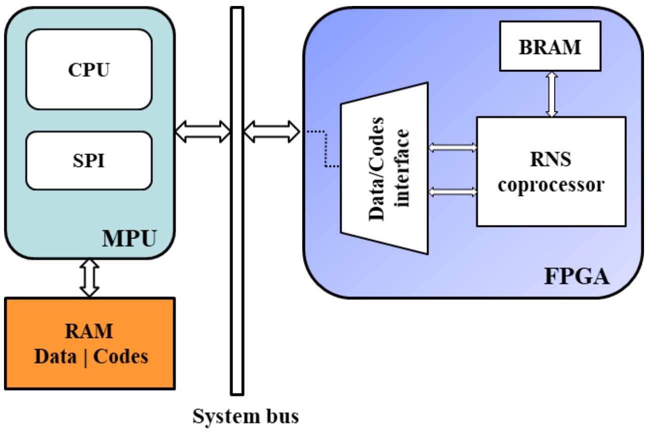


Figure 5.1 Diagram of hardware platform for ECC operations evaluation

5.2 HW performance

Point Multiplication evaluation

The PM defined in Alg. 4.1 (see section 4) was evaluated for the NIST prime curve P-256 defined in [1]. The test vector used for the curve P-256 is defined in [2]. The PM was coded through a Java tool emulator (see chapter 2). Annex A illustrates the interface of the emulator and the generated files including the microcode and data for the RNS operation.

An embedded software was coded to provide the microcode and data to the RNS coprocessor through the SPI ports. The control signals from the RNS interface module were handled through interruptions.

The Table 5.1 provides a PM performance comparison among some state-of-the-art ECC processors and the RNS coprocessor evaluated in this thesis. The RNS coprocessor and the one in [3] use both 16-bit data paths, while MicroECC processor [4] uses 32 bits. These variants were selected among the others considering their comparable device occupancy and the type of FPGA used, although they are not exactly the same. In all the cases, the cycles count is referred to the PM with a 256-bit scalar.

Apart from the performance, from the security point of view, the RNS coprocessor is the only one in the Table 5.1 that incorporates an intrinsic protection against correlation attacks, due to the parallelism of the channels operation. Moreover, the randomization of the inputs in the Alg. 4.1 gives additional resilience to such attacks. The SPA protection in the RNS processor is given by the use of the Montgomery ladder method. The processor in [3] incorporates a SPA countermeasure through the use of Montgomery ladder as well, but there is not mention to protection against CPA. The MicroECC has the same sort of protections, not including CPA. Recall that a horizontal correlation attack against ECC has proven to be successful as shown by Bauer et al. in [8].

Table 5.1. Comparison of PM performance among state-of-the-art ECC processors

	PM cycles count	**Area (slices)**	**BRAM**	**Freq. (MHz)**	**Device**
RNS coprocessor (this work)	2 601 547	944	15	88,3	FPGA Spartan-6 (xc6slx75)
ECC coprocessor [3]	3 227 993	1832	9	108,2	FPGA VirtexII-Pro (xc2vp30)
MicroECC [4]	949 951	1158	3	210	FPGA VirtexII-Pro (xc2vp7)
ECC coprocessor [5]	--	1704	--	225	Virtex-7

On the other hand, in [5], the FPGA device used has a superior technology than the one used in this work. That impedes to make a fair comparison to our work. Still, the area occupancy of the RNS is almost the half. Recall that the RNS is a generic arithmetic processor which can be used for other purposes, while the architecture in [5] is optimized to perform ECC operations. It is expected that the implementation of the RNS coprocessor in one of the most advanced FPGA, improves the PM performance.

The work in [6] presents a RNS-based ECC processor for PM. The hardware architecture is based on the straightforward binary method for Point Multiplications, which is vulnerable to SPA and Timing Attacks. Therefore, the performance of this processor is not considered for comparison in this thesis.

Multiple Point Multiplication evaluation

The MPM operation was evaluated in the RNS coprocessor using the same hardware platform configuration.

Contrary to the PM (if using Montgomery ladder), the MPM execution time strongly depends on its joint scalars; it is, depending on the $\mathrm{i}\text{-}th$ bits of all the scalars, the computation is more or less complex (see Section 4.2). Besides, the NAF method used depends on the hardware resources, specifically the number of precomputed values, and that impacts on the performance of the operation.

The previous facts make difficult to do a performance comparison of this operation. For this reason, the performance of MPM in this work is given in terms of the number of Montgomery multiplications, as it can be seen in Section 4.2

5.3 Conclusions

In this chapter the HW platform to conduct the evaluation of the ECC Point Multiplication and Multiple Point Multiplications is provided. Due to the characteristics of the latter, its performance is already provided in the section 4.

Regarding the PM, a comparison is done to other similar works. The comparison is relatively fair, although the devices are not from the same family, but their technologies are close. A proper comparison with the work in [5] is difficult to do because it presents a recent ECC processor implemented in a more advanced technology. However, the RNS implementation is more compact, which is desirable for area constrained devices.

The security features of the RNS coprocessor exceed the others, as they are not protected against CPA. The operation of each of the 16 channels is a source of noise to each other. This acts as a natural countermeasure against a CPA. Additionally, it avoids embedding a source of noise, like a random number generator, as it is commonly done in secure embedded devices.

References

1. Digital Signature Standard (DSS). Federal Information Standards Processing Publication 186-2, National Institute of Standards and Technology, 2000.
2. Cryptographic Algorithm Validation Program. Test vectors. https://csrc.nist.gov/projects/cryptographic-algorithm-validation-program/digital-signatures
3. J. Vliegen et al. A compact FPGA-based architecture for elliptic curve cryptography over prime fields. ASAP-2010, 21st IEEE International Conference on Application-specific Systems, Architectures and Processors, Rennes, 2010.
4. M. Varchola, T. Güneysu, O. Mischke. MicroECC: A Lightweight Reconfigurable Elliptic Curve Crypto-processor. International Conference on Reconfigurable Computing and FPGAs, 2011.
5. D. Amiet, A. Curiger, P. Zbinden. Flexible FPGA-Based Architectures for Curve Point Multiplication over GF(p). Euromicro Conference on Digital System Design (DSD-2016), 107-114, 2016.
6. D. M. Schinianakis, A. P. Kakarountas, T. Stouraitis. A new approach to elliptic curve cryptography: an RNS architecture. IEEE Mediterranean Electrotechnical Conference (MELECON), 2006.
7. Silica Xynergy-M4 Board flyer. https://dsp-sys.de/files/pdf/xyzbayxynergy_board_flyer_2012.pdf
8. A. Bauer, E. Jaulmes, E. Prouff, and J. Wild. Horizontal collision correlation attack on elliptic curves. Selected Areas in Cryptography, LNCS 8282, 553–570, Springer, 2013

Conclusions

This thesis has been oriented to the implementation of an authentication scheme that guarantees being efficient enough for low-power devices, and also secure to resist some of the most harmful physical attacks. In this sense, a suitable authentication scheme was identified. The fundamentals of the underlying cryptography involved was studied and its potential vulnerabilities. The Side Channel Attacks that poses a threat to the related cryptographic algorithms were identified as well as the countermeasures to overcome such attacks. A modification was introduced to a Point Multiplication algorithm which allows a more efficient implementation. Also, an efficient hardware technology was chosen for the implementation which also has an intrinsic resilience to the said attacks. Finally, the implementation of the critical cryptographic operations was evaluated.

Bellare's BNN-IBS was found as a suitable authentication scheme as it avoids the use of expensive certificates. A modest modification was introduced to better adapt the scheme to the RNS coprocessor without losing its efficiency. Moreover, this IBS could be implemented in the coprocessor still applying the SCA countermeasures for a more complete protection.

Regarding the ECC operations in which the authentication is based, an improvement that reduces the co-Z Montgomery ladder algorithm was introduced in chapter 4. This algorithm saves 4MM in the point multiplication and avoids the computation of the y-coordinate, which also implies the avoidance of the expensive transformations from the

Montgomery domain, a further transformation from RNS to binary. The y-coordinate could be avoided specially in the verification phase of the BNN-IBS due to the introduced modification. Additionally, the load of microcode to the processor was devised as potentially vulnerable to a SCA that aims at recovering the secret scalar's bits. A well-known masking solution was suggested to mitigate such threat.

The evaluation of the Point Multiplication was conducted in the HW platform built around a FPGA-based RNS coprocessor, given the fact that its architecture allows efficient and secure implementation of cryptographic ECC-based algorithms. A comparison with other similar works was conducted, taking into consideration the technology used. The small footprint of the implementation is highlighted compared to the others, while the PM performance is comparable too, acknowledging that this coprocessor is a generic one, not specifically tuned for cryptographic operations.

The RNS coprocessor does not compute divisions, however, that operation is involved in the authentication schemes implemented. The need to solve the division problem motivated the study of modular inversions. From the security analysis of some algorithms, two of them were found potentially vulnerable, and the SCA to exploit a naïve implementation of them are described herein.

Future work

Another PM approach

In [1] a Point Multiplication technique is introduced which seems more efficient than the one used in this work. A research in that sense could be conducted to evaluate the feasibility of its implementation in the RNS coprocessor, given the mentioned benefits of the RNS architecture.

Bilinear pairings

Bilinear pairings technique is a rather new research line which would improve the implementation of certificate-less authentication algorithms. Its performance in a FPGA-based platform is recently demonstrated [2]. However, these algorithms are vulnerable to SCA as well [3]. An interesting topic of future research would be the evaluation in terms of performance and security of bilinear pairings in the RNS coprocessor.

Security evaluation

The inherent noise due to the parallel channels operation in the RNS coprocessor, along with the applied countermeasures in the PM should avoid SPA, DPA and CPA attacks, but this fact has not been confirmed in this work. An interesting complementary work would be to conduct a security evaluation of the RNS coprocessor during the PM computation.

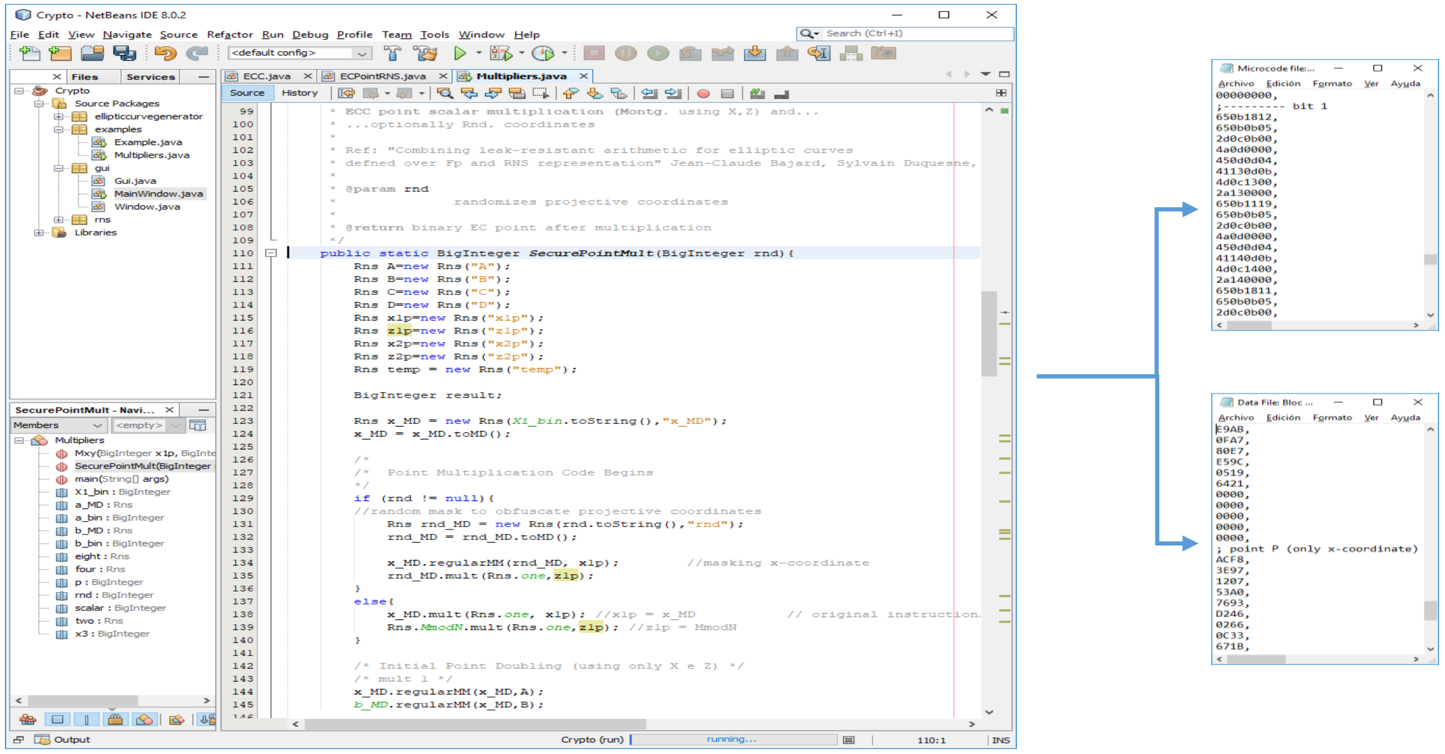
[1] M. Morales-Sandoval and A. Diaz Perez. Novel algorithms and hardware architectures for Montgomery Multiplication over GF (p). Laboratorio de Tecnologías de la Información., Tech. Rep., 2015.

[2] Z. Hao, W. Guo, J. Wei and D. Sun. Dual Processing Engine Architecture to Speed Up Optimal Ate Pairing on FPGA Platform. 2016 IEEE Trustcom/BigDataSE/ISPA, Tianjin, 2016.

[3] Jauvart, D., El Mrabet, N., Fournier, J.J.A. et al. Improving side-channel attacks against pairing-based cryptography. J Cryptogr Eng 10, 1–16, 2020.

Annex A

Java tool for the emulation and configuration of the RNS coprocessor



```
     * ECC point scalar multiplication (Montg. using X,Z) and...
     * ...optionally Rnd. coordinates
     *
     * Ref: "Combining leak-resistant arithmetic for elliptic curves
     * defned over Fp and RNS representation" Jean-Claude Bajard, Sylvain Duquesne,
     *
     * @param rnd
     *              randomizes projective coordinates
     *
     * @return binary EC point after multiplication
     */
    public static BigInteger SecurePointMult(BigInteger rnd){
        Rns A=new Rns("A");
        Rns B=new Rns("B");
        Rns C=new Rns("C");
        Rns D=new Rns("D");
        Rns x1p=new Rns("x1p");
        Rns z1p=new Rns("z1p");
        Rns x2p=new Rns("x2p");
        Rns z2p=new Rns("z2p");
        Rns temp = new Rns("temp");

        BigInteger result;

        Rns x_MD = new Rns(X1_bin.toString(),"x_MD");
        x_MD = x_MD.toMD();

        /*
        /*  Point Multiplication Code Begins
        */
        if (rnd != null){
        //random mask to obfuscate projective coordinates
            Rns rnd_MD = new Rns(rnd.toString(),"rnd");
            rnd_MD = rnd_MD.toMD();

            x_MD.regularMM(rnd_MD, x1p);          //masking x-coordinate
            rnd_MD.mult(Rns.one,z1p);
        }
        else{
            x_MD.mult(Rns.one, x1p); //x1p = x_MD            // original instruction
            Rns.MmodN.mult(Rns.one,z1p); //z1p = MmodN
        }

        /* Initial Point Doubling (using only X e Z) */
        /* mult 1 */
        x_MD.regularMM(x_MD,A);
        b_MD.regularMM(x_MD,B);
```

```
00000000,
;--------- bit 1
650b1812,
650b0b05,
2d0c0b00,
4a0d0000,
450d0d04,
41130d0b,
4d0c1300,
2a130000,
650b1119,
650b0b05,
2d0c0b00,
4a0d0000,
450d0d04,
41140d0b,
4d0c1400,
2a140000,
650b1811,
650b0b05,
2d0c0b00,
```

```
E9AB,
0FA7,
80E7,
E59C,
0519,
6421,
0000,
0000,
0000,
0000,
0000,
; point P (only x-coordinate)
ACF8,
3E97,
1207,
53A0,
7693,
D246,
0266,
0C33,
671B,
```

Annex B

Template Attack against RSA key generation based on BEEA

In this section, a Template Attack against a protected BEEA implementation is introduced by the first time. This TA targets the RSA key generation, however, it could also be conducted against an ECDSA implementation where the target would be the modular inversion.

For the feasibility of this attack, the followings assumptions are made:

- The hardware device employed for the TA has a source of leakage
- The target RSA key generation uses BEEA to compute the secret key d and/or $GCD(e, x-1)$ for coprimality tests.
- The BEEA implementation is either unprotected or uses the Chartier's masking method in eq. 3.6.
- An adversary is able to acquire enough power (or electromagnetic) traces from the target device (preferred) or similar devices running BEEA
- An adversary is able to acquire a power (or electromagnetic) trace during the target RSA key generation phase

Usually, once an RSA key pair is generated, it is stored to be used several times. In the case of the coprimality tests, they are performed only once involving p and q (the two final successful prime candidates). Something similar occurs in ECDSA with the nonce k. These facts impose a restriction, as the target secrets are used only once. Therefore, the proposed attack is specifically a Single Trace Template Attack. Nevertheless, due to the nature of the Euclidean algorithm and $\phi(N)$ in RSA, at least eight attack points can be targeted in a single trace to obtain eight leakage vectors. Thus, the procedure herein is still formally an STA but equivalent to an 8-trace TA, where the success probabilities are higher.

If it is denoted by p_{AP} the probability of a successful attack on a single attack point, then

$$p_{STA} = 1 - (1 - p_{AP})^8$$

would be the probability of a successful attack on the key generation process. Furthermore, if an adversary is able to acquire n traces of the key renewal procedure, the probability of a successful attack increases, and could be defined by

$$p_{nSTA} = 1 - (1 - p_{AP})^{8n}$$

The profiling phase of the TA is currently unfeasible on 32-bit devices (employing modest resources) because of the amount of generated data and large computations, although for 16-bit devices it could still be possible. In this work we consider the proposed attack on 8-bit devices; however, it could be extended to 16-bit devices.

Attack points on the private key generation

In the case of the private key generation, the v-loop is executed first, and at least twice, because $\phi(N)$ is divisible by four. The operation flow for the first and second iterations could be easily predicted. The table 1 shows the ordered execution of the relevant operations involved.

Table 1. Target operations into the *v-loop* of BEEA (e, φ(N))

	Operation 1	Operation 2	Result 1	Result 2
1st Iteration	v = v / 2	C = (C + φ(N)) / 2	v = φ(N) / 2	C = φ(N) / 2
2nd Iteration	v = v / 2	C = C / 2	v = φ(N) / 4	C = φ(N) / 4

* *C* = 0 at the beginning of the computation (see Alg. 2.2)

All these operations (assuming a common implementation) perform a right shift followed by a copy-to-register. Furthermore, the only data being manipulated is the secret $\phi(N)$, which is shifted one and two bits in the first and second iteration respectively. This is an advantage for TA, because the leakage of both: the shift and the copy-to-register operations, is uniquely related to the value of ϕ(N).

The fact of manipulating $\phi(N)$ and $\phi(N)/2$, should not be an issue to consider all the operations as equivalent attack points. The guessed bytes in the second iteration shifted one bit left, should correspond to those guessed in the first iteration; or what it is the same: the guessed bytes in the first iteration shifted one bit right should correspond to those guessed in the second iteration. Therefore, operations 1 and 2 for the first two iterations of $BEEA(e, \phi(N))$ could be considered as four equivalent attack points into the same power trace, from where four leakage vectors can be obtained.

Notice that, if $\phi(N)$ has more than two trailing zeroes, it would be easily detected in the power trace, because the power profile of the following iterations (into the v-loop) would be quite similar to the previous ones. If further leakage vectors are extracted, the guesses should be done as in the table 2. Thus, it can be said that, the more trailing zeroes in $\phi(N)$, the higher would be the probability of a successful attack.

Table 2. Target operations into the *v-loop* of BEEA (*e*, *φ*(*N*)) beyond the 2nd iteration

	Operation 1	Operation 2	Result 1	Result 2
3rd Iteration	v = v / 2	C = (C + φ(N)) / 2	v = φ(N) / 8	C = 5 · φ(N) / 8
		C = C / 2		C = φ(N) / 8
4th Iteration	v = v / 2	C = (C + φ(N)) / 2	v = φ(N) / 16	C = 13 · φ(N) / 16
		C = C / 2		C = φ(N) / 16

* *C* = *φ*(*N*) / 4 at the beginning of the 3rd iteration (see table 1)

** Because e might be randomized (as in eq. 2.2), *D*`s evenness cannot be predicted, and the conditional branches into the *v-loop* cannot be predicted either (although they might be distinguished in the power trace). That is why we give the two possible results from the operation 2.

Experimental environment

To apply the attack against a BEEA implementation, a number of measurements of the power consumption is acquired, which includes the operation of Table 1. The target is an 8-bit BEEA algorithm. The power consumption traces are collected during 50,000 runs on the ChipWhisperer-Lite power collection board at a sampling rate of 29.54Ms/s. To construct various templates for the secret value $\phi(N)$ is selected as a random 128-bit value which is divisible by 4. The technique employed to perform the attack is based on [1].

Experimental results

The attack was performed using one power trace, and the average success rate was calculated by repeating it 1,000 times for each case. The task of choosing one out of 5,000 traces and performing the attack was repeated 1,000. As a result, each 8-bit value of the secret $\phi(N)$ is recovered with a probability of more than 98.90% in a single-trace attack.

[1] E. Özgen, L. Papachristodoulou and L. Batina. Template attacks using classification algorithms. 2016 IEEE International Symposium on Hardware Oriented Security and Trust (HOST), McLean, VA, 2016.