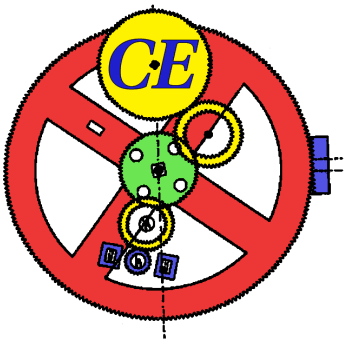

Computer Engineering
Mekelweg 4,
2628 CD Delft
The Netherlands
http://ce.et.tudelft.nl/

2010

MSc THESIS

Security in RFID systems

Dimitris Stafylarakis

Abstract

The thesis addresses the topic of security and privacy for RFID systems. The current state of the art on RFID technology is presented, along with a brief overview of the physical principles governing such systems. A description of the concepts of security and privacy and potential problems in RFID systems due to lack thereof follows. Within this framework, two components are presented and proposed as solutions for the aforementioned problems. The first is a hardware/software platform for privacy protection of end users of RFID systems. It selectively jams the electromagnetic signals used for communication between components of an RFID system, thus preventing undesired access to private data. The second component is a circuit that can be integrated in RFID chips in order to enhance their level of security. It generates a digital signature using properties of elliptic curves that cannot easily be reverse engineered, thus offering trustworthy means of authenticating the hosting chip. Both components take the attributes of a typical RFID system into account, in order to demonstrate the feasibility of implementing acceptable solutions for existing RFID systems.

CE-MS-2010-12

TU Delft

Faculty of Electrical Engineering, Mathematics and Computer Science

Security in RFID systems

Theory and practice for enhancing security in RFID systems

THESIS

submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE

in

COMPUTER ENGINEERING

by

Dimitris Stafylarakis
born in Athens, Greece

Computer Engineering
Department of Electrical Engineering
Faculty of Electrical Engineering, Mathematics and Computer Science
Delft University of Technology

Security in RFID systems

by Dimitris Stafylarakis

Abstract

The thesis addresses the topic of security and privacy for RFID systems. The current state of the art on RFID technology is presented, along with a brief overview of the physical principles governing such systems. A description of the concepts of security and privacy and potential problems in RFID systems due to lack thereof follows. Within this framework, two components are presented and proposed as solutions for the aforementioned problems. The first is a hardware/software platform for privacy protection of end users of RFID systems. It selectively jams the electromagnetic signals used for communication between components of an RFID system, thus preventing undesired access to private data. The second component is a circuit that can be integrated in RFID chips in order to enhance their level of security. It generates a digital signature using properties of elliptic curves that cannot easily be reverse engineered, thus offering trustworthy means of authenticating the hosting chip. Both components take the attributes of a typical RFID system into account, in order to demonstrate the feasibility of implementing acceptable solutions for existing RFID systems.

Laboratory : Computer Engineering
Codenumber : CE-MS-2010-12

Committee Members :

Advisor:	Georgi Gaydadjiev, CE, TU Delft
Chairperson:	Koen Bertels, CE, TU Delft
Member:	Wouter Serdijn, ELCA, TU Delft
Member:	Stephan Wong, CE, TU Delft

Dedicated to my parents

Contents

List of Figures

List of Tables

Acknowledgements

Learn from yesterday, live for today, hope for tomorrow. The important thing is not to stop questioning. - Albert Einstein

Thanks to everyone that taught me something in the process.

Dimitris Stafylarakis
Delft, The Netherlands
June 15, 2010

1 Introduction

Modern times exhibit extremely fast-paced technological developments. New technologies are incessantly introduced to the modern person which in turn must adopt them, become familiar with them and incorporate them in his everyday life. From active research topic to a fully-grown market, RFID technology has already followed an impressive trajectory and can be found in a large number of real-life applications. The purpose of this thesis is to contribute in further developing RFID technology, in order to serve its purpose in a better way.

1.1 A first look into RFID

RFID stands for Radio Frequency Identification and the corresponding technology is concerned with contactless data-carriers and their usage in various applications. The contactless property of RFID can be extremely practical in some applications, yet considerably challenging to apply. This is mainly due to its interdisciplinary nature, which brings together elements of as diverse fields as RF technology, semiconductor technology, data protection and cryptography, telecommunications, manufacturing technology, juridical sciences and public management and administration.

The range of applications is equally diverse; security/access control, asset management, transportation, supply chain management, point of sale, toll collection, automobile immobilizers, baggage handling, animal tracking, real time location systems are very few of the application areas where RFID systems are put to use, with new ideas constantly appearing. Similar technologies include *Barcode* systems, *Optical Character Recognition* (OCR) systems, *biometric* systems and smart cards. RFID technology offers competitive properties when compared to all these technologies. They require less complicated hardware than OCR systems, they cost less and are faster than biometric systems and they are reprogrammable, as opposed to barcode systems. The competitiveness of RFID technology is also proven by the immediate popularity it has received so far. In particular, the RFID market value is currently estimated at $2.77 Billion and is expected to grow six times until 2010 [22]. The advantages of RFID technology include low manufacturing costs, minimal operating and mainentenance costs, operation from a distance up to 5m, reasonably fast access rates and storage of changeable data.

RFID Systems comprise one or more *transponders* or *tags* and one or more *readers*. Tags are the actual data-carrying devices, equipped with a microchip and an antenna, which is used as a coupling element to a reader. Readers contain an antenna, the coupling element to tags, a control unit and typically some kind of interface to a host PC (e.g. RS232, USB). The host PC may run a data management system to make use of the information sent by tags, according to the specific application requirements. In principle, the tag is activated only when it is located in a specific range from the

reader, which defines the *interrogation zone*. The reader supplies the synchronization signals and initiates communication with the tag by issuing *reader requests*. The tag subsequently must respond using appropriate transmission procedures via *tag responses*. In many cases, the tag is also powered by the reader; the latter transfers power to the former through the antennas by electromagnetic coupling.

1.1.1 Classification of RFID devices

There are numerous variants of RFID devices. RFID systems operate either in *half/full duplex mode*, where the RF field of the reader is constantly switched on, or *sequentially* where the RF field is briefly switched off for the tag to transmit its response. RFID tags can have a memory capacity as small as one bit (e.g. in *Electronic Article Surveillance* Systems [19]) or larger, programmable or not, using different memory technologies.

Furthermore, tags are classified as *passive* or *active*, depending on the way they are powered. Passive tags are powered by the electric/magnetic field produced by the reader within its interrogation zone, whereas active tags include a battery unit that supplies all or part of the necessary power. Passive tags are naturally cheaper both to operate and to maintain. On the other hand, active tags have a larger power budget and usually larger operating range or enhanced processing and storage capabilities. However, active tags are not suitable for placement in unaccessible spots as battery replacement is an arduous task. A hybrid form of tags uses a battery to power up the digital part of the tag but not the communication part of the tag.

Another distinctive feature of tags is their operating frequency. Operating frequencies utilized by RFID systems span a large part of the frequency spectrum used for communications today and are divided in *low frequency* LF (30-300 KHz), *high frequency* HF (3-30 MHz), *ultra high frequency* UHF (300 MHz-3 GHz) and *microwave* ($>$ 3 GHz). The electromagnetic radiation emitted by RFID devices should by no means disrupt the operation of other radio systems. For this reason, the suitable range of frequencies for such systems, as well as their output power is regulated by standards. Frequencies in ISM bands (reserved for Industrial, Scientific and Medical applications) are therefore used in most of the cases. Operating distances depend on the operating frequency and are differentiated in *close-coupling* (e.g. ISO 10536), *remote-coupling* (e.g. ISO 14443 and ISO 15693) and *long range systems* (e.g. ISO 18000-6). Close coupling systems are coupled with both electrical and magnetic fields, they operate in the low and high frequency ranges and have typical operating ranges of (0 - 1cm). Remote coupling systems are the most popular systems currently in the market, they are typically based on inductive (magnetic) coupling, they operate in the high frequency range and have a typical operating distance of (1m). Long range systems usually operate using electromagnetic waves in the UHF or microwave frequency range. Reading distances can reach 6-7 meters. An indicative chart of the various standards and their characteristics can be seen in Figure 1.1. The simplest ones (EAS) have a memory capacity of just a few bytes and a basic functionality of reading from and writing to the memory. The most complicated ones are essentially smart cards augmented with RFID functionality. Their memory capacity can reach 128 KBytes. This type of cards include more advanced features such as cryptographic operations and are therefore considerably more expensive.

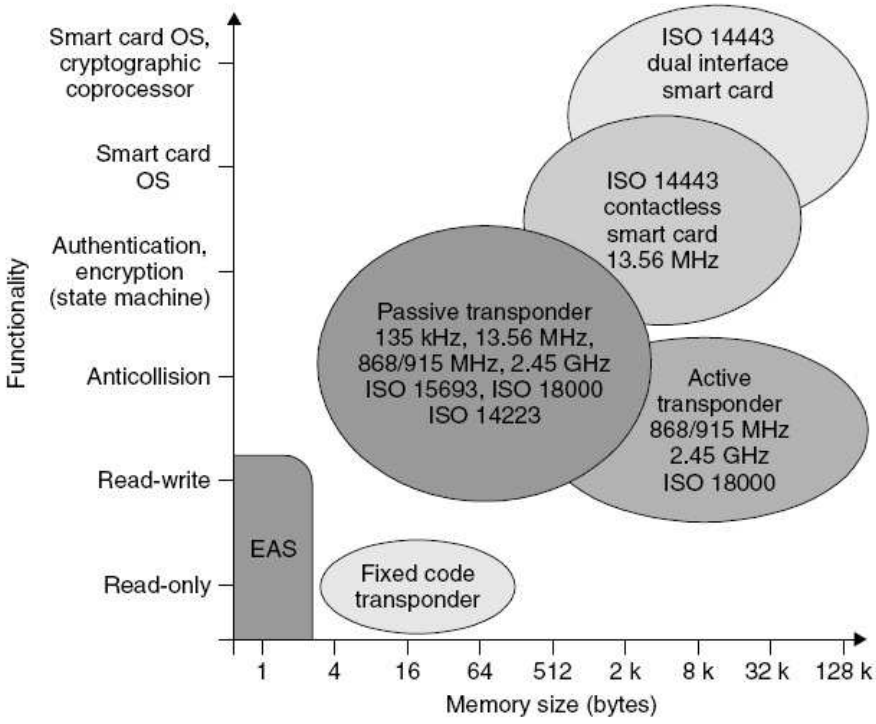

Figure 1.1: Classification of RFID systems according to their functionality

Finally, differentiation of RFID systems can be based on the way that tags send their data back to the reader, which can be one of the following:

1. use of reflection or backscatter, where the antenna of the tag reflects part of the wave back to the reader. The frequency of the reflected wave is the same as the transmission frequency of the reader. This technique is used mainly for systems working in the UHF band.

2. load modulation, where the tag directly influences the reader's field and is used in low and high frequency systems, at frequencies below 30 MHz. Load modulation will be presented in detail in 2.1.1.

3. the use of subharmonics and the generation of harmonic waves in the tag. This technique is usually chosen for low frequency systems and hardware implementations are relatively simple.

1.2 Introduction to Security Engineering

The concept of security has been introduced in systems where malice, error or mischance by its users may incur serious consequences. Critical systems such as nuclear plants cannot afford any possible security breach; electronic transaction systems need to guarantee

their users safe transactions. Security requirements greatly differ from one system to the other and the system designer must ensure that critical sub-systems are properly protected. Omitting security provisions can be catastrophical for the system and the system end users. Emphasizing too much on them can render the system inconvenient in the best case and oppressive in the worst one. In every case, however, the environment surrounding the system evolves and security-oriented tools and methods must adjust to this evolution in order to maintain reliability.

A security policy formalizes the proper and improper use of a system, the possible threats against it and countermeasures against these threats. In practical applications, such countermeasures are referred to as *security objectives*. A security protocol defines a set of steps that must be followed by two or more communicating parties, in order to satisfy the desired security objectives. Security objectives can be one or more of the following:

- *Confidentiality*: The secrecy of the exchanged data is maintained by the communicating parties.
- *Authentication*: The origin of a message can be verified by the receiver.
- *Integrity*: The receiver of a message can ascertain that the message was not altered in transit.
- *Non-repudiation*: The sender of a message should not be able to deny later having sent it, and this should be verifiable independently by a third-party without knowing the message contents.

A more detailed description of the concepts behind security objectives and security engineering in general can be found in [9]. These security objectives are desired for digital transactions to acquire the element of trust. Mutual trust is fundamental for e-commerce and e-government applications that have recently become very popular, for both the corporate/government and the customer/citizen sides. Its existence is essential in order not to hold up economic development or efficient public administration.

Several *security mechanisms* can assist in achieving security objectives. A set of such mechanisms is mentioned in the security architecture of the Open Systems Interconnect model [41], some of which are applicable also in other systems:

- Encryption or Encipherment;
- Digital Signature mechanisms;
- Access Control mechanisms;
- Data Integrity mechanisms;
- Authentication exchange mechanism;
- Traffic padding mechanism;
- Routing control mechanism;

- Notarization mechanism.

Encryption is the process of obscuring information to make it unreadable without special knowledge. In order to achieve this, suitable algorithms, termed *ciphers*, are used. According to one definition, a *digital signature* is an electronic sound, symbol, or process, attached to or logically associated with a contract or other record and executed or adopted by a person with the intent to sign the record. A more specific definition is given in Section 2.4. *Access Control* determines which users and system elements can access which resources of the system. Access control is of central importance in electronic systems and the corresponding mechanisms span all different system levels, from the operating system to the application level. *Data Integrity* means preserving data during storage or transport operations, for their intended use, while ensuring an expected quality level relative to specified operations. *Authentication exchange* is a mechanism intended to ensure the identity of an entity by means of an information exchange. *Traffic padding* is concerned with generating spurious data units during communication to disguise the amount of real data units being sent. These mechanisms generally belong to the cryptology discipline which comprises two branches, cryptography and cryptanalysis, the former concentrating on designing ciphers and the latter on breaking them.

All these mechanisms are not necessarily restricted to algorithmic (software) constructs. *Biometrics* identify people by measuring some aspect of individual anatomy or physiology, some behavioral characteristic (such as the handwritten signature) or a combination of these. *Tamper-resistant hardware* utilizes special barriers, sensors and alarms in order to prevent unauthorized entities from gaining access to critical information storage resources of the system. In extreme cases, such resources are located in reinforced buildings, guarded by security personnel. These cases are mentioned in order to indicate the importance of security in modern systems, which also justifies the considerable amounts of money invested in it.

1.3 Thesis Overview

The purpose of this thesis is to consider possible applications of security mechanisms in future RFID technologies. It is by no means exhaustive, nor universal, as the complexity of the topic, the diversity of security aspects and the perpetually changing state-of-the art in RFID technology (as well as technology on the whole) render such an attempt prohibitive. Nevertheless, this work hopes to contribute, through hardware implementations of specific algorithms, towards investigating the applicability of certain security mechanisms in future RFID devices.

Chapter 2 presents one aspect of improving security in RFID systems. Chapter 2.2 lays the ground by introducing inductively-coupled RFID cards and relevant protocols in detail. Examples of security weaknesses are then presented, which signify the necessity of security-enhanced RFID systems, with emphasis put on privacy and authenticity protection. In Section 2.3, a platform that assists in privacy protection for RFID tags is presented. Finally, a brief introduction on digital signature schemes and elliptic curves can be found in 2.4.

The subject of the following chapters is the design and implementation of signature generating hardware, according to the Elliptic Curve Digital Signature Algorithm (ECDSA). The feasibility of such functionality within a severely resource-constrained device such as an RFID tag, is further investigated. Chapter 3 serves as an introduction to the concepts of elliptic curve cryptography as well as the ECDSA standard in particular, covering all the necessary mathematical background in order for the reader to become familiar with this complex topic. In addition, a number of algorithms that perform the desired elliptic curve cryptographic operations is given, as well as some additional considerations beyond the selection of the algorithm itself, that ensure a secure implementation. Chapter 4 describes the implementation of the hardware in concern. The design objectives are set in 4.1.1. Principles of proper hardware design are presented briefly, to explain the design choices that are made for the unit. Subsequently, information on the actual design itself is presented (Section 4.5) and the results concerning its performance are compared to relevant work (Section 4.7). Finally, Chapter 5 sums up the presented work and concludes the thesis.

RFID Tags and security of Infrastructure

2

2.1 Inductively-coupled contactless vicinity cards

In this section, an overview of inductively coupled RFID systems is given, as well as some specifics of a representative standard ISO 15693 [6]. This will provide some insight on the technical details of a typical RFID system. ISO 15693 is selected as a standard with wide commercial application and therefore of significant importance. All the research and work in the context of this thesis is targeted for this specific standard. This, however, does not limit the suitability of the proposed techniques to other RFID standards or other application contexts. Tags compliant with ISO 15693 typically belong to the low- to mid- end side in the RFID tag taxonomy (as seen in Figure 1.1) and thus are more resource constrained and have less capabilities when compared to devices compliant to other standards. As a result, hardware design for such devices is challenging and requires great effort. An in-depth treatment of RFID systems in general, including the specific standard, can be found in [19]. The information presented in the following paragraphs is mostly based on this book.

2.1.1 Principle of operation

RFID tags of the vicinity-card type, are almost always passively operated. The reader device generates a strong electromagnetic field at a frequency of 13.56 MHz (termed the *carrier frequency* f_c). When an LC resonant circuit is moved into the vicinity of the reader's field, a voltage induced by the field will appear on the circuit's terminals. If the resonant frequency of the circuit is equal with f_c, then this effect will be maximized (*sympathetic oscillation*). The antenna of an RFID tag serves the purpose of the LC resonant circuit. Since the wavelength of the field is several times larger than the distance between the reader and the tag, the coupling field can be considered to be a simple alternating magnetic field. The coupling between a reader and a tag can then be viewed as a transformer, with the primary coil belonging to the reader and the secondary one belonging to the tag. The *coupling coefficient* k represents the quality of the magnetic coupling between the two coils. The system is based on the weak magnetic coupling of tag and reader coils. Tags brought too close to the reader cannot function properly, as the coupling coefficient becomes larger.

As is known from transformer analysis, the current induced at the tag's coil counteracts the original phenomenon, which is noticeable as a small voltage drop at the reader's coil and a corresponding weakening of the generated field. This voltage drop can be quantitively controlled by changing the current flowing in the tag's coils in a controlled way. It is then detectable by the reader and this phenomenon forms the channel for data communication within such RFID systems with passively-powered tags. Moreover, the

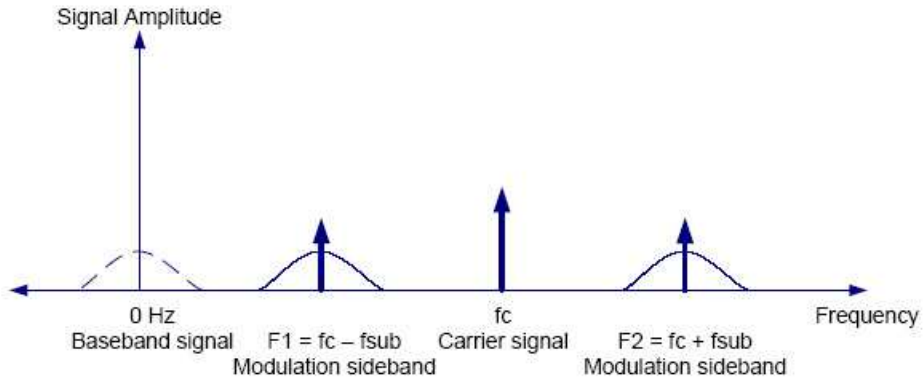

Figure 2.1: Frequency spectrum resulting from load modulation with subcarrier technique

induced voltage is rectified and serves as a power supply for the tag.

This approach differs from standard telecommunication techniques in the sense that in the latter, the antennas of the reader and the tag are considered as transmitting and receiving energy, instead of being magnetically coupled. In order to achieve proper energy transfer at the operating frequency of this system, large antennas would have to be used. According to antenna theory, the antenna length (if an ideal dipole is considered) would have to be $\sim 5m$, one quarter of the wavelength of the transmitted electromagnetic field (which for a frequency of 13.56 MHz would be $\lambda \approx 22m$). While this is possible for large ground-stationed readers, such dimensions are prohibitive for portable tags. For higher frequencies, smaller antennas can be used. UHF RFID systems usually have a longer range of operation, because their small antennas are more effective in their operating frequency.

The efficiency of the power transfer depends on many different factors and this is the principal reason for the difficulty in achieving efficient communication between a reader and a tag. A tag and a reader cannot be studied separately, but viewed as an entity. Factors affecting the efficiency of power transfer include the quality Q of the coils (where $Q = f_c/\Delta f$, Δf being the frequency band where an antenna can transmit without corrupting the signal), the operating frequency, the ratio of windings of the 'primary' and 'secondary' coils, the relative position of one coil with respect to the other and the area enclosed by the tag's coil. For higher frequencies, the number of windings reduces and smaller antennas can be used.

Data transfer from a tag to the reader uses the *load modulation with subcarrier* technique. Load modulation corresponds to varying the tag's circuit parameters in time with the data stream, in order to influence the transformed impedance of the secondary coil (tag) in the transformer equivalent. This results in a voltage measured at the terminals of the primary coil (reader) which varies in sync with the data stream. Subsequently, this voltage is sampled and the transmitted data stream can be thus reconstructed from the reader's circuitry. The data to be transmitted does not modulate the carrier signal

directly. Instead, they modulate a *subcarrier*, which is a signal of frequency f_{sub}. The derived signal subsequently modulates the carrier (of frequency f_c, an integer multiple of f_{sub}). The result of this technique is the generation of two sidebands at a distance of $\pm f_{sub}$ from f_c, which contains the actual data (the resulting signal spectrum is similar to the one shown in Figure 2.1). Usage of sidebands is essential because the voltage variations induced on the received signal on the reader are significantly weaker when compared to the carrier signal itself. As a result, complex circuitry would be required to retrieve data modulated at the carrier signal directly, whereas by using sidebands a simple filter is required to discard carrier noise.

2.1.2 The ISO 15693 standard

The standard itself consists of three parts:

- **Physical Characteristics** describing the physical attributes of the vicinity cards used in the system.
- **Air Interface and Initialization** specifying details on the magnetic field properties as well as the data encoding and modulation procedures.
- **Anticollision and transmission protocol** defining the commands, the initialization sequence for communication between cards and readers, methods to detect and resolve collisions when multiple cards attempt to communicate to one reader and finally application-specific options to optimize operation.

According to the standard, readers initiate transactions with tags and determine the mode of communication. Mode selection and the exact command are specified in the reader request, which has special fields for this purpose. The tag response contains the requested information or an error indication. Tags have a memory capacity of 8 kBytes at maximum. Physical memory is organized in up to 256 blocks of up to 256 bits each. For instance, the ICODE SLI tag (manufactured by NXP Semiconductors) contains a maximum of 1 kbit memory. Single or multiple memory blocks can be read by the reader using appropriate commands. More importantly, each tag stores a unique 64-bit identifier internally, to distinguish between tags. This identifier contains information on the manufacturer as well as the tag itself. Last, but not less important, the standard specifies timing limits within which the tag must respond, otherwise its response is discarded. From the moment the reader has completed transmitting the request, a tag must complete all internal calculations and start transmitting its response within $[318.8, 323.3](\mu sec)$. This limitation imposes hard real-time requirements to the tag subsystem, which deeply affect further design choices. The restricted time budget, combined with the bounded processing capabilities of the tag require careful design in order to meet the requirements.

A lower-complexity design lowers manufacturing costs. Therefore, it is desirable as RFID tags are meant to be low-cost devices. In fact, a company called PolyIC announced (in 2006) the production of printable RFID tags with minimal capabilities. The project is still in its development stage, but is a clear indication about the next generation of RFID tags ([43]). Moreover, the data carrier chip must remain at low costs. The manufacturing

costs of chips are affected by the percentage of faulty chips during production, as well as the area occupied by the logic on the die. As a result, chip designers attempt to keep the design as simple as possible and to minimize the required area. RFID tags usually support simple functions (such as accessing the memory and transmitting its contents) so the required manufacturing costs are very low. For tags supporting more advanced functionality, such as cryptographic operations, the cost rises, but still must be kept as low as possible, to make secure tags appealing for massive production and deployment.

The distance between a tag and the reader, such that there is just enough energy for the tag to operate is called *energy range* of the transponder. However, for the whole system to be functioning, the maximum range is such that the response of the tag can be properly detected by the reader. The weaker the field strength at the location of the tag, the lower the power budget of the tag. A specific current I at the antenna of the reader generates a corresponding field strength around the antenna. The minimum sufficient field strength H_{min} is related to the maximum distance between the reader and tag antennas with the Equation (2.1). N_1 and R are the number of windings on the antenna of the reader and its radius respectively. H_{min} depends on the frequency of the carrier generated at the reader, the shape of the reader's antenna and the electronics used at the analog front-end of the tag.

$$x = \sqrt{\sqrt[3]{(\frac{I \cdot N_1 \cdot R^2}{2 \cdot H_{min}})^2} - R^2} \tag{2.1}$$

The mathematical derivation of the formulas that calculate the energy range of a typical ISO 156930 compliant tag, as well as the rest of the involved parameters can be found in [19]. Figure 2.2 is reproduced from the same book and shows how the energy range of a tag is affected by the current consumption (and consequently by the power consumption) of its circuitry. The dotted line represents this relationship when the tag supply voltage can be 3V and the solid line represents a supply voltage of 5V. Current consumption in the range [1, 10] μA (corresponding to [5, 50] μW for 5V supply voltage) does not affect the energy range. Beyond this limit, the energy range deteriorates fast. For minimal power consumption, the energy range is [1, 1.5] meters for typical tags of this category.

The standard defines a number of modes of operation for both the tags and the readers, suitable for noisy environments and other application requirements. Considering communication from the reader to the tag, the former initiates the communication by sending requests to the latter. The structure of the request message is specified in the standard. The data payload is encoded with a pulse position modulation scheme, i.e. the time position of a pulse corresponds to a specific data value. Two encoding schemes are specified and the choice depends on the capabilities of the reader. In the first one, one symbol is mapped to one byte (*1 out of 256*, Figure 2.3) and in the second one two bits (*1 out of 4*, Figure 2.4). In the explanatory figures, the byte $(225)_{16}$=$(01001011)_2$ is shown, using each encoding scheme. The former scheme divides the transmission symbol in 256 slots and the outgoing byte is encoded by assigning a logical low pulse to the appropriate slot. This results in a data rate of 1.65 kbps. The latter scheme uses 4 symbols to transmit one byte. Each symbol is divided in 4 slots and the low pulse's position encodes 2 bits, which results in 26.48 kbps. The selected mode of operation

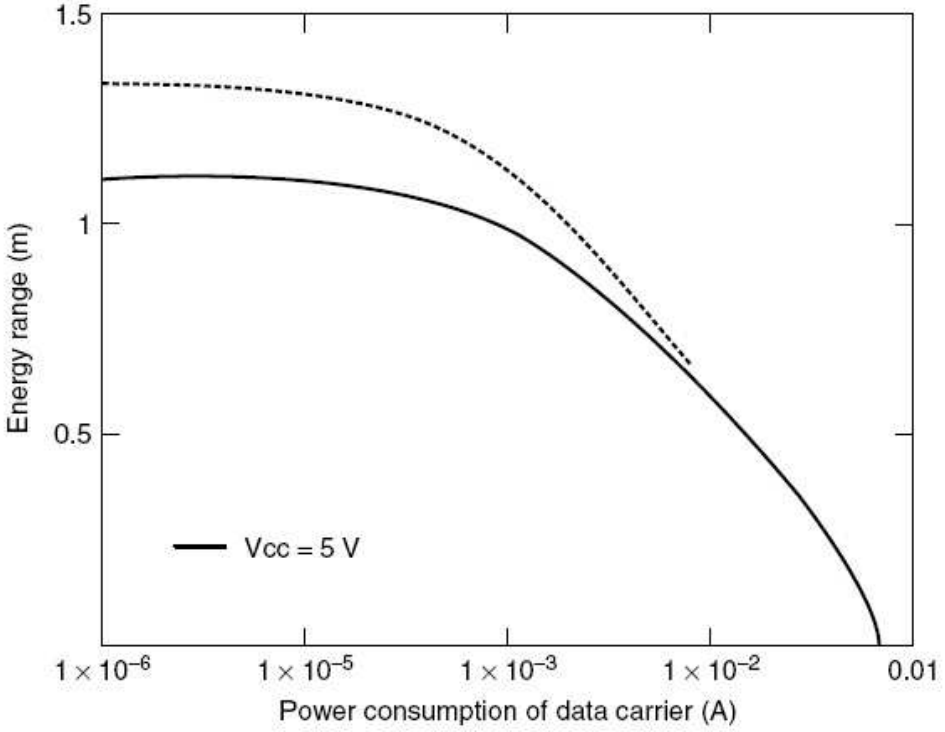

Figure 2.2: Energy range of a tag with respect to its power consumption.

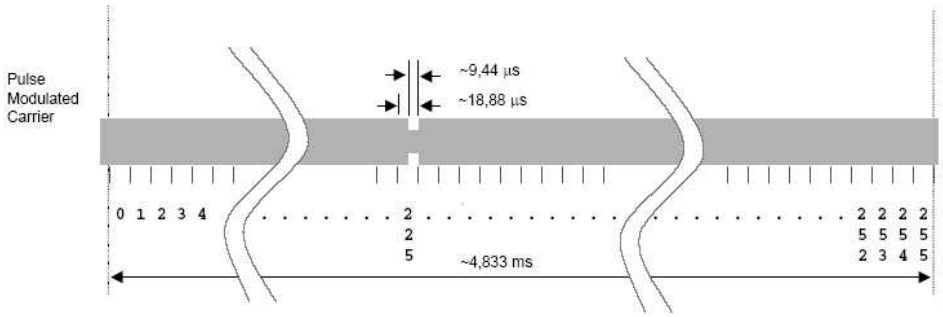

Figure 2.3: Representation of 225 in 1 out of 256 encoding

is indicated to the receiving tags by use of pre-defined Start of Frame (SOF) headers. The encoded symbol is then used to modulate the amplitude of a carrier of 13.56 MHz frequency, using either 10% ASK or 100% ASK. ASK stands for *Amplitude Shift Keying* and varies the amplitude of the carrier signal to distinguish between the two logical values. In the case of 10% ASK, a logical high corresponds to 100% of the carrier amplitude and a logical low to (100 - 10)% = 90% of the carrier amplitude. In the case of 100% ASK the respective values are 100% for logical high and absent carrier for logical low. The latter case is more noise-resistant, it reduces however the received power on the tag, hence narrowing the power budget for the tag operations themselves.

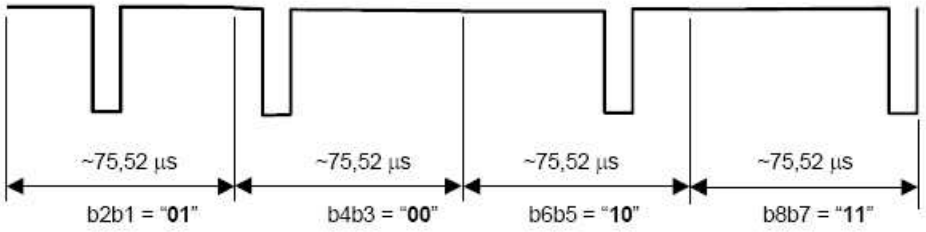

Figure 2.4: Representation of 225 in 1 out of 4 encoding

Concerning communication from a tag to a reader, the available options are again targeted for different operational environments and tags of different capacity. The particular options used in an application are specified by the reader, therefore a tag complying to the standard must support all specified options in order to communicate with every possible reader. The payload of the tag response is encoded using the Manchester Encoding scheme, where a logic 0 is represented by a negative pulse edge (transition from logical high to logical low) and a logic 1 is represented by a positive pulse edge. The Manchester encoded signal is then used to modulate the so-called subcarrier signal, to form the baseband signal. A subcarrier is a signal that is used to modulate the main carrier signal. The standard states that either one subcarrier is used, or two subcarriers. The two schemes, along with the exact timing specifications of the standard, are depicted in Figure 2.5 and Figure 2.6 respectively. In both cases, a logical 0 is depicted, which is assigned to a high half-bit followed by a low half-bit. When one subcarrier is used, the high half-bit is represented by 8 pulses of frequency $f_{sub1} = f_c/32$=423.75 KHz followed by the same time period with no signal (ASK). When two subcarriers are used, the high half-bit is represented by the same pulses of f_{sub1} and the low half-bit is represented by 9 pulses of frequency $f_{sub2} = f_c/28$=484,28 KHz (*Frequency Shift Keying* or FSK, where each logical value is assigned to a different frequency). It should be noted that the subcarrier signals defined by the standard are derived by dividing the carrier frequency by an integer, so that the tag can obtain its clocking signals directly from the input signal.

Again, two different data rates are possible. To simplify implementations, the two schemes are identical except for the timing of the half-bit period. The signals as depicted in the figures achieve a data rate of $\sim$ 26.5 kbps. A lower data rate is possible if the half-bit period is quadrupled. The final signal that is fed to the output is derived by amplitude modulation of the carrier signal.

2.2 Necessity for security provisions in RFID applications

Security in RFID applications can have different meanings, depending on the specific application needs. At this moment, security provisions in RFID standards are in their infancy and universal specifications, that would render RFID technology secure and trustworthy, do not exist yet. One reason for this is that security is not always required in RFID systems. For instance, systems for tool recognition do not have such requirements,

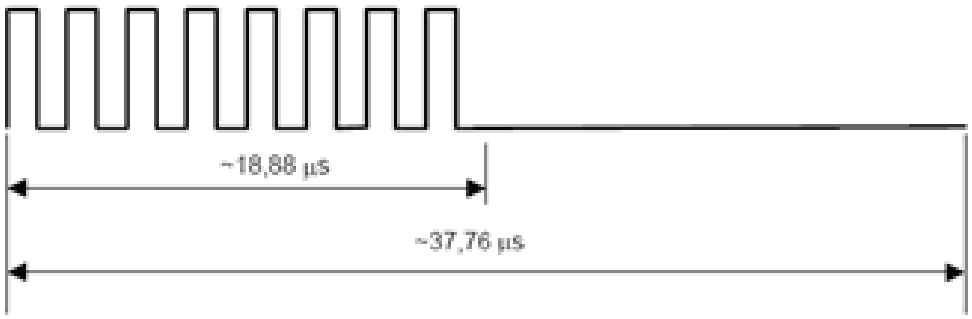

Figure 2.5: Representation of Logic 0 using a single subcarrier.

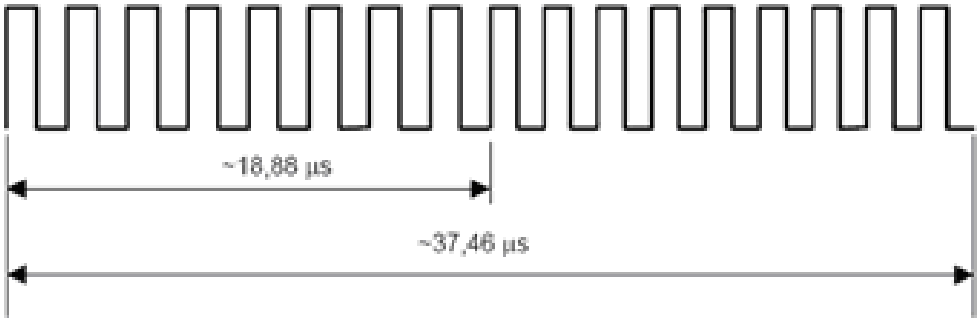

Figure 2.6: Representation of Logic 0 using two subcarriers.

in most of the cases. Extra security mechanisms are in such cases redundant and therefore they unnecessarily increase the cost. On the other hand, lack of provisions for securing an RFID system might eventually cause severe troubles to ignorant users, when potential threats exist.

Attacks on RFID systems may take one of the following forms, according to [19]:

- Unauthorized reading of a data carrier in order to duplicate or modify its data.
- Placement of a foreign data carrier within the interrogation zone of a reader with the intention of gaining unauthorized access to a building or receiving services without paying, and
- Eavesdropping into radio communications and replaying the data, in order to imitate a genuine data carrier.

In order to show possible RFID security threats more clearly, some more detailed cases are presented.

Threats against Privacy: As RFID applications become unceasingly more widespread, RFID tags will be issued with all sorts of items surrounding people and their everyday activities. They are commonly classified as unobtrusive electronics, which

means that a person carrying RFID tags may not be aware of their presence. For instance, tagged clothing articles may still contain a functioning RFID tag, even after purchase. Unpermitted access by third persons can introduce a privacy threat for the carrier of the RFID tag. The least harmful versions of such threats are embodied in retailers keeping track of their customers' shopping behavior and preferences, for statistical reasons. In worse cases, an inventory of personal items can easily be obtained with the help of RFID technology. Appropriately equipped individuals may keep track of a person's location using combinations of RFID readings, hence reviving the "Big brother" threat and raising questions of trust in the consumer community [8]. Committees have been formed to look carefully into the matter and long debates have been triggered by it. It is therefore clear that RFID protocols should be reconsidered to allow for protection against privacy threats, in the context of the confidentiality security objective of Section 1.2.

Threats against Authentication: One of the emerging applications of RFID technology is RFID-equipped passports. RFID tags are already in use in access control applications, in major corporations and governmental buildings. An increasing number of countries, both in the European Union and overseas, provide (or are considering to do so) citizens with passports that include an RFID tag containing personal information such as name, date of birth and a digitized version of the passport photograph [20]. However, research groups have managed to reproduce passport data and clone passports [45] and thus point out another possible security threat related to RFID technology. In order to satisfy the authentication, integrity and non-repudiation security objectives of Section 1.2, digital signature schemes have been proposed for similar RFID applications and are seriously considered by the involved authorities.

All the above issues make it worthwhile for the scientific community as well as RFID manufacturers and users to look carefully into the matter of enhancing RFID systems with the privacy and security elements that are missing in the current standards. Users will cease to mistrust RFID technology and this will significantly accelerate the penetration of RFID in everyday life. Scientists, on the other hand, face the challenge of incorporating some computationally intensive components into devices with rigid restrictions, as far as resources (e.g. chip area and power consumption) are concerned. Accordingly, possible solutions are herewith investigated for the aforementioned privacy and security issues of currently used RFID standards.

2.3 RFID tags and Privacy

One problem, as stated earlier, is concerned with preventing RFID Readers from accessing the contents of RFID tags in the proximity of the user, without implicit or explicit consent from the user. An extreme approach is to make use of killing or sleeping commands that are specified in RFID protocols (also in ISO 15693), which make RFID tags cease functioning altogether or temporarily respectively. This approach is neither desirable nor practical in most of the cases. For example, killing the tag prevents also the legitimate user from using it in a profitable way. Furthermore, a "sleeping" tag could be awoken by an unauthorized reader, thus offering curtailed protection. A better alternative is to selectively permit a Reader to access the tag's contents, following some

authentication procedure. RFID tags will not respond to unauthorized readers' requests.

In this context, a platform for testing such functionality was developed and is presented in this thesis. The platform was developed to be used by the *RFID Guardian* system [46]. It concerns a battery-powered device, integratable into Personal Digital Assistants (PDA) or mobile phones, which is carried by users to manage the security and privacy against threats that attempt to exploit the surrounding RFID tags. The RFID Guardian utilizes existing RFID protocols to facilitate the following security-related services: auditing, access control, key management and authentication. It acts as an intermediate between RFID Readers and tags, monitors requests to tags by external readers and accordingly permits a tag response to reach the reader.

The contribution of this work concerns the design and implementation of the hardware platform supporting this functionality, as well as the device driver for operating it through a PDA or a cellphone processor. The platform will provide an interface to both RFID Readers and RFID tags, compliant with the ISO 15693 standard. A block diagram of the main components of the device is depicted in Figure 2.7. The heart of the device is a microcontroller, which controls the analog front-end. The latter comprises a circuit to communicate with RFID tags (emulating a reader), a circuit to communicate with RFID readers (emulating a tag) and the necessary voltage translating circuits as an interface to the microprocessor.

In order to communicate with an RFID Reader, the prototype includes circuits emulating a tag's functionality. According to the theoretical description of the previous paragraphs, the tag receiver is essentially an amplitude demodulator, a circuit well known to analog designers. The tag receiver's design (Figure 2.8) is based on the *SA605 FM mixer system* from Philips ([48]), configured as an ASK demodulator. The chip itself contains a mixer/oscillator, two limiting Intermediate Frequency (IF) amplifiers, a logarithmic Received Signal Strength Indicator (RSSI) and a voltage regulator internally. The input signal is translated to the IF of 10.7 MHz, for which the internal circuits of SA605 are optimized. The mixing stage of the IC is used for this purpose, along with a crystal oscillator, externally implemented. Subsequently, the signal is amplified using the IF amplifier of the IC. Finally, the demodulated signal is obtained by the RSSI circuitry, which follows the envelope of the received signal. Proper signal filtering between the stages of the chip is performed by means of ceramic filters, which are centered around 10.7 MHz. The RSSI output is led to a data slicer, which restores the binary signal, filtering out spikes due to ambient noise to prevent them from distorting the signal.

The data slicer is a simple comparator, with rail-to-rail output capabilities. Signal filtering is performed via a low-pass RC filter (formed by R_7 and C_{47} in Figure 2.9), which allows the data signal to pass through and filters out noise coming from the RSSI output. A low-pass RC filter uses a combination of a resistor R and a capacitor C which allow only frequencies below a threshold ($f_{cut} = 1/RC$) to pass through. This threshold is chosen so that the baseband signal (after demodulation) remains intact and high-frequency noise is left out. In addition to that, a dynamic-threshold level is maintained, for the data slicer to decide whether the incoming signal is a logical high or not. This is necessary because as the device moves around within the interrogation zone of the reader, the incoming signal's strength changes accordingly. Therefore, the incoming signal is compared to this dynamically adjusting threshold, which is obtained

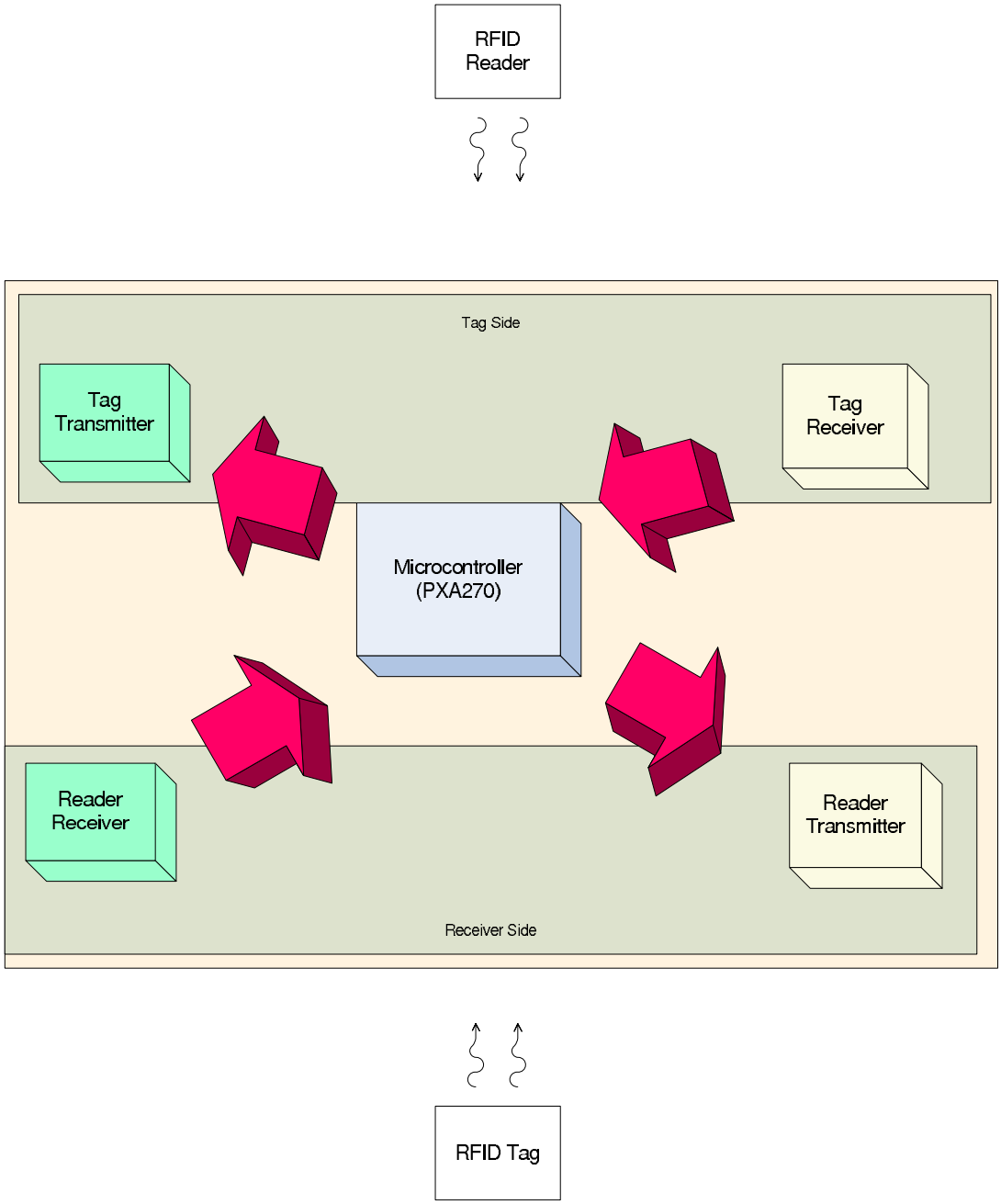

Figure 2.7: Block Diagram of the platform for the RFID Guardian

via a second low-pass filter (R_4 and C_{53} in Figure 2.9), with a cut-off frequency much lower than the one of the first low-pass filter. It essentially operates then as an averaging circuit, producing the average of the incoming signal. The time constants of the two RC filters were obtained empirically, until an acceptable signal could be seen at the output

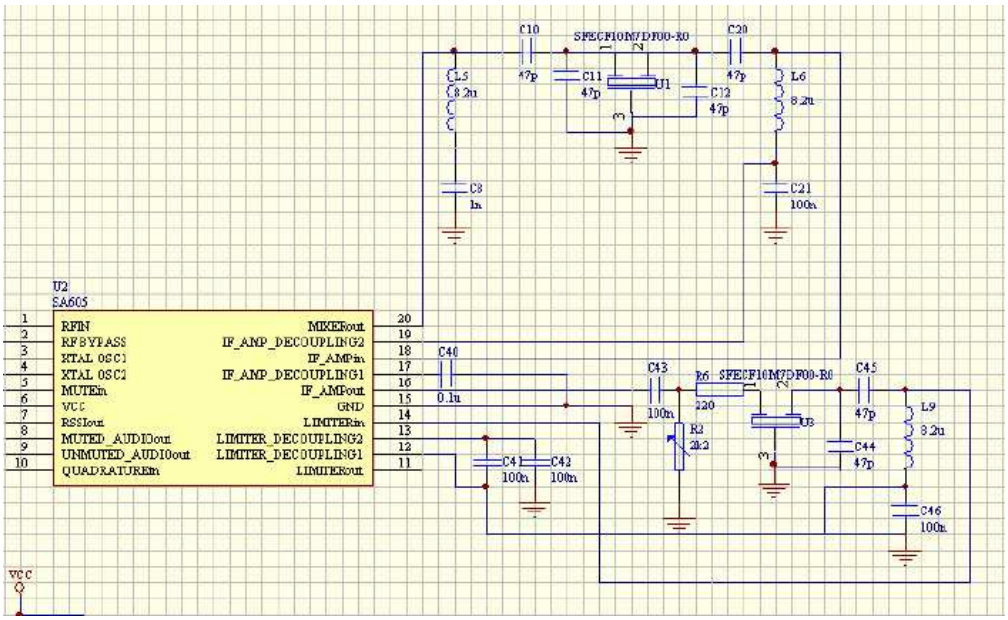

Figure 2.8: The SA605 chip and the IF filtering circuitry

of the data slicer, while moving the device around the interrogation zone. A detailed analysis of the demodulator, can be found in [39]. The data slicer circuit is shown in Figure 2.9.

The clean signal is led to the microcontroller, which is responsible for decoding the baseband signal in order to obtain the original payload from the reader. The decoding procedure, which is software based, is simple counting of the time distance between pulses, which is unique for each symbol. The original data value mapped to the symbol is then extracted in a straightforward manner.

A component of major importance is the tag transmitter. It is desirable for the transmitter to produce a signal that is strong enough in order to communicate with readers as distant as possible from the device. Naturally, the receiver must be adequately sensitive in order to receive signals from such distant readers. This combination will guarantee a maximum protection from potential privacy threats. The fact that RFID Guardian is a battery-operated device can be used to its advantage, as it can be designed to emulate an active tag. As such, it generates the carrier signal on board (using the Pierce oscillator of Figure 2.10) and modulates it with the baseband signal for transmission. The generated carrier signal is a square wave of 13.56 MHz frequency and all the modulation circuitry, forming the waveforms defined by the standard, are supported by standard CMOS logic circuits, hence keeping costs and implementation complexity low. Frequency dividers are used to generate the subcarrier signals from the carrier and the rest of the signal-generating logic is used to select the required subcarrier, depending on the selected baseband encoding scheme and the data to be encoded. The baseband signal is then sent to the mixer (which for digital signals is a simple AND gate) to modulate the carrier. The modulated signal is subsequently led to the power amplifier stage. This is a class D amplifier with adjustable duty cycle of the generated pulses, for accurate control of

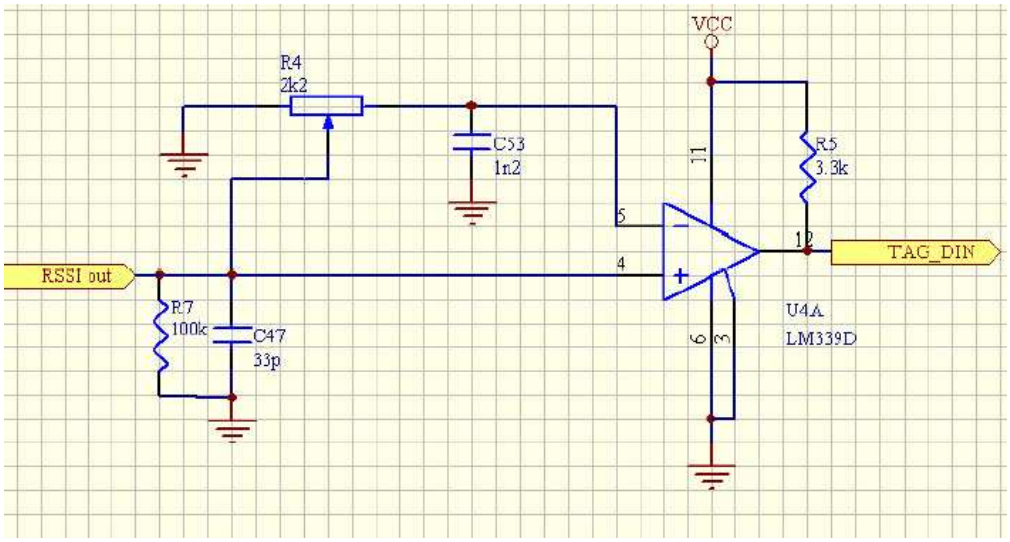

Figure 2.9: Data slicer circuit

the output power. It is also composed of low-cost standard CMOS logic, based on the design presented in [34] and in Figure 2.11.

Baseband encoding is performed with the aid of the microcontroller. Depending on the chosen encoding scheme, a synchronization pulse, generated by the tag transmitter, interrupts the current process in fixed time intervals, so that the microcontroller can update the transmitter control signals in time, based on the data bit to be transmitted next. The synchronization signal is asserted every half-bit period of the Manchester-encoded data signal. This configuration allows the microcontroller to determine the value of the transmitted message with a half-bit resolution, which is useful for generating special messages, as will be explained in the following paragraph. The transmitter circuits subsequently generate the corresponding waveform. A more robust implementation would be based on the transmitter circuitry alone, without intervention of the microcontroller. With the current implementation, it is important that the software routines running on the microcontroller are in sync with the transmitter hardware. This is because a possible delay in the software routines could result in a delayed response to the synchronization pulse and subsequently to corrupted waveforms. This risk exists especially with multitasking operating systems which cannot guarantee that control will be returned to the device driver process within the required time limits. Utilization of a small low-cost microcontroller, running the device driver algorithm alone would be a viable option, if component count is not a restriction.

Device components and the driver software where initially designed with the aid of an *AVR Mega32* microcontroller from ATMEL [10], in order to focus on debugging the device driver's code without concern for the underlying operating system or application-level algorithms involved with the RFID Guardian's functionality. Subsequently, the platform was integrated with a *PXA270* microprocessor from Intel [24] running a real-time operating system. In specific, a port of eCos [1] for this specific processor runs as a basis for developing the device driver as well as the application layer software

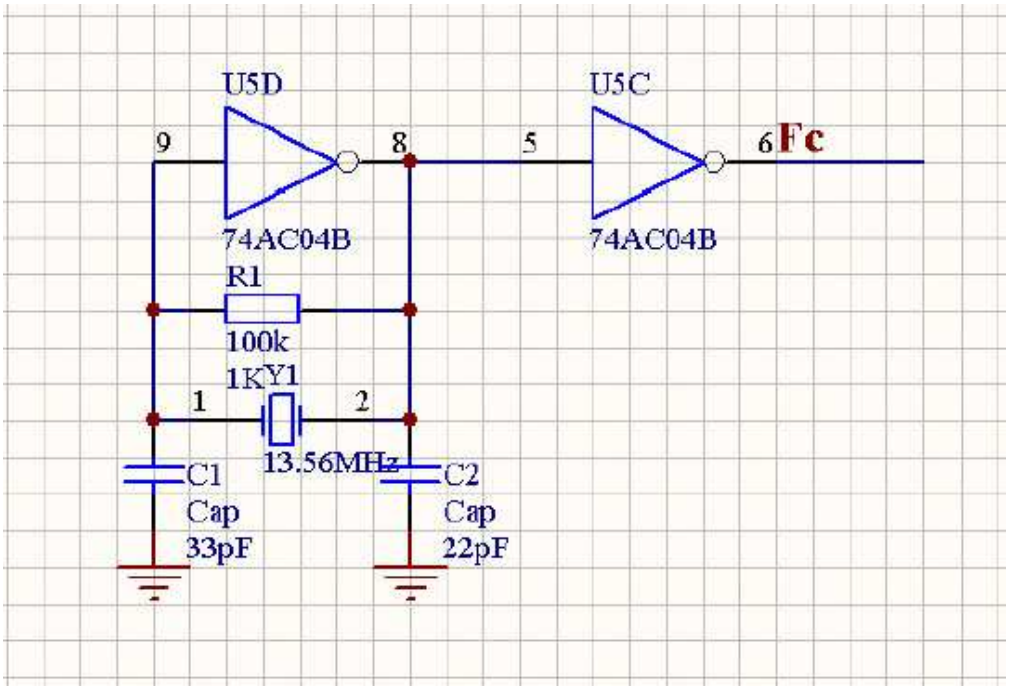

Figure 2.10: Pierce oscillator with a 13.56 MHz crystal

components.

As mentioned above, the tag side of the device must emulate a tag's functionality. Furthermore, it is responsible for generating jamming signals, in order to prevent unauthorized readers from communicating with RFID tags in the vicinity of the Guardian. Since the transmitted signal is determined by the microcontroller on a half-bit basis, it is fairly easy to produce output signals that do not comply with the waveforms specified in the standard. For instance, two consecutive logical high half-bits are not allowed in a legitimate Manchester encoded message. These signals are superimposed to the actual tag responses and block the reader, as they are considered to be garbage or corrupted responses. The success of this approach is based on the properties of amplitude demodulation, which is used by ISO 15693 compliant RFID Readers. Amplitude demodulation only considers the amplitude of the incoming signal, discarding phase and frequency information. A continuous jamming signal could also successfully jam an RFID reader, however this could be easily detectable by a slightly more sophisticated reader. Our device, on the other hand, produces signals that could only be perceived as corrupt tag responses. The microcontroller runs a high-level algorithm that determines whether a reader should be jammed or not. One way of determining when such a jamming should be used, is by maintaining an access control list internally, which contains identification numbers of readers that are authorized to access tags in the vicinity of the user. Tag and reader identification numbers are related in the access control list, so a reader may access a specific subset of the existing tags. More information can be found in [46]

The reader side is implemented as a fully functional RFID reader. It is based on Melexis 90121 RFID Transceiver [33], which is compliant with a number of RFID Stan-

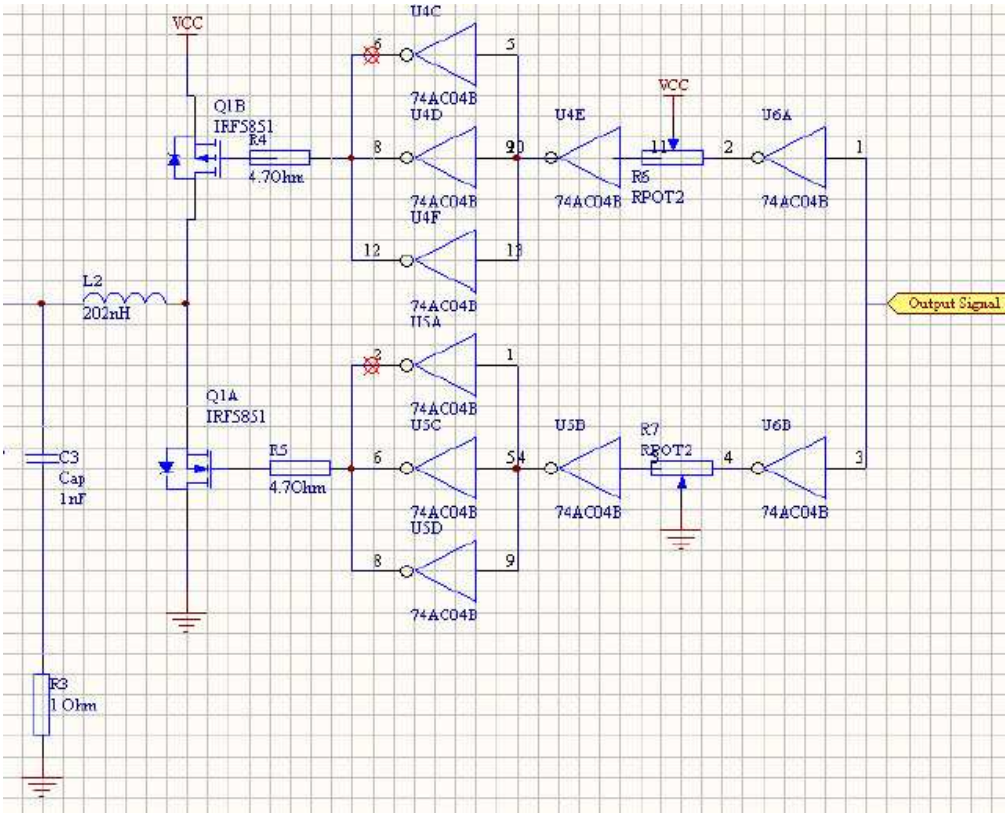

Figure 2.11: Class D power Amplifier

dards (including ISO 15693). It is capable of handling outgoing reader requests as well as incoming tag responses, leaving only a minor encoding/decoding task to the microcontroller (Figure 2.12). The chip can be directly interfaced to the microcontroller through a simple serial protocol. Regarding the output stage, the chip includes a power amplifier stage of relatively small output power, which is circumvented and replaced by a 5W power booster identical to the one used for the tag transmitter (Figure 2.11). This results in an increased operating range for the device.

The complete schematics are shown in Figure 2.16 (Reader side including the MLX90121 IC), Figure 2.14 (tag receiver using the SA605 IC) and the custom-made tag transmitter in Figure 2.15. The resulting prototype is depicted in Figure 2.13. Its correct functionality was tested and verified with a commercial RFID Reader (SL RC400) and tags, from the ICODE SLI line of Philips. The reader side, the tag transmitter and the tag receiver have been implemented on separate modules for testing. A single-loop antenna, which is wound using a CD case (to provide an appropriate form-factor), is used to perform the actual transmission. In order to achieve proper impedance matching (for better output power transfer from the power amplifier to the antenna), some fine tuning of the matching elements is required. In order to produce acceptable signals, a lot of factors have to be taken into account, especially proper design of grounding and power supply circuits. A ground plane was used whenever ground noise reached unacceptable

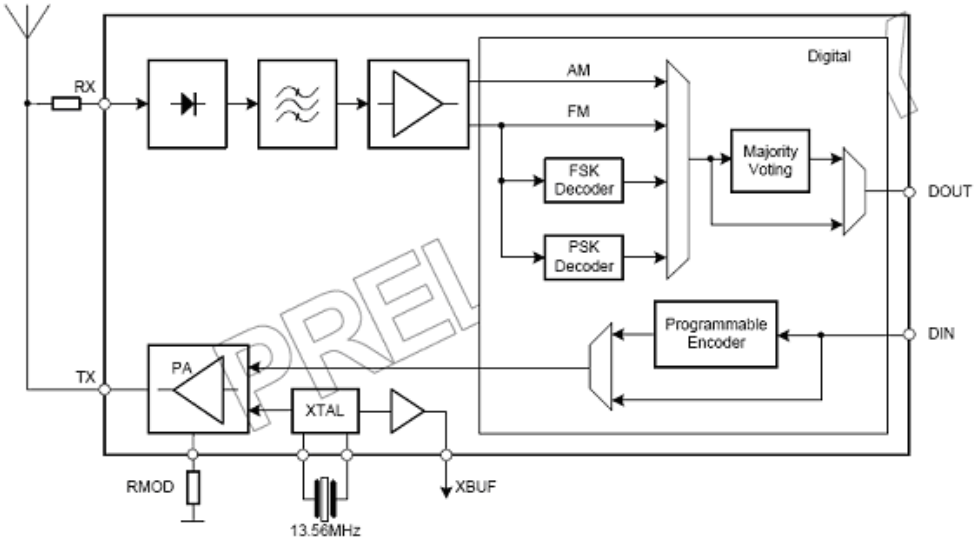

Figure 2.12: Melexis MLX90121 Block Diagram

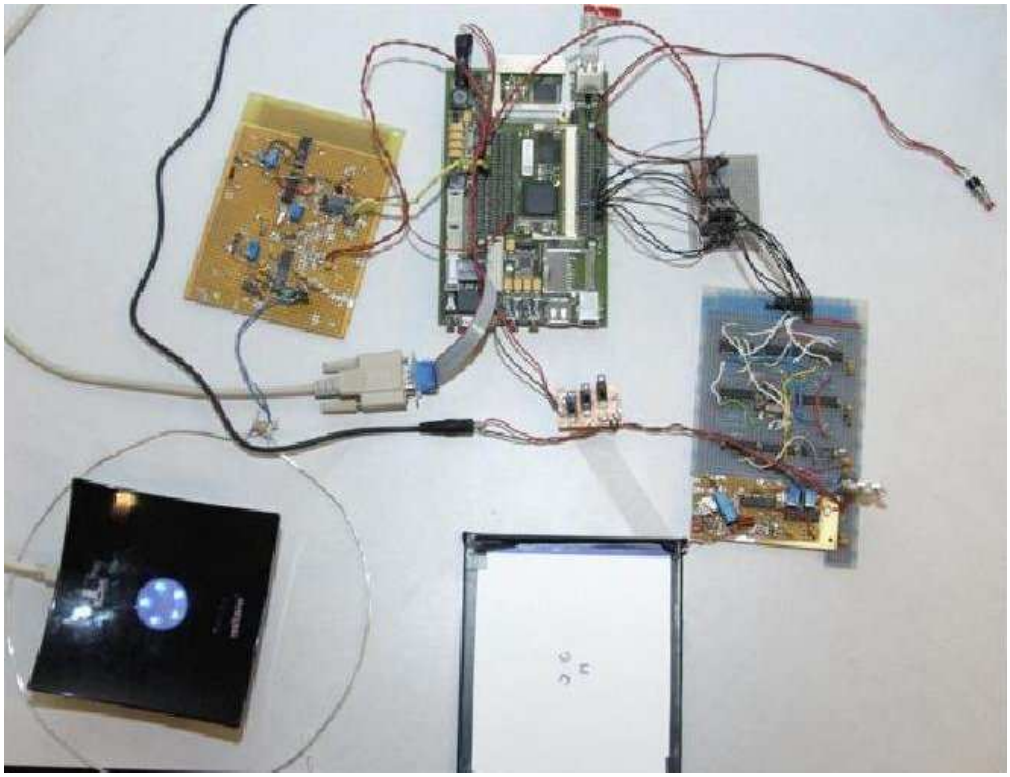

Figure 2.13: RFID Guardian Prototype

levels, especially in the power amplification circuits. In addition, decoupling capacitors were amply used, to filter out noise coming from the power rail.

2.4 Digital Signatures for RFID tags

In Section 2.2 the necessity for enhanced security capabilities in RFID tags used in access control applications was pointed out. The main focus is on digital signatures and the possibility of integrating them in RFID applications.

European law (directive n.93/1999) defines two different kinds of electronic signatures, with increasing juridical value; *electronic signature* and *advanced electronic signature*. The first one, described as *means data in electronic form which are attached to or logically associated with other electronic data and which serve as a method of authentication*, is solely used for authentication purposes. Advanced electronic signatures must additionally meet the following requirements:

- They are uniquely linked to the signatory;
- They are capable of identifying the signatory;
- They are created using means that the signatory can maintain under his sole control;
- They are linked to the data to which they relate in such a manner that any subsequent change of data is detectable.

Advanced electronic signatures guarantee the integrity of the associated data and non-repudiation of the signing act. They ensure that the signature's secret data cannot be retrieved by a third person within a reasonable amount of time, i.e. the validity period of the signature generating device. This is the type of signature with the highest juridical value. More details can be found in [2].

Digital Signature Schemes fit in the general context of Public Key Infrastructure (PKI). They exploit one-way functions (mathematical operations that are much easier to calculate in one direction than in the reverse direction) and their properties in order to perform the required cryptographic operation. One of the desirable properties of 1-way functions is the existence of a *trapdoor*, which simplifies the reverse operation by knowledge of specific data. Symmetric cryptographic algorithms, though easier to realize and more efficient in operation, have the disadvantage that a secure channel between all participating parties needs to be established, before actual communication commences. This is impossible in many practical signing applications. As a result, Public Key cryptographic algorithms are used, such as the RSA [47], based on multiplication of large integers, the DSA [38] based on modular exponentiation and the ECDSA [7] based on scalar multiplication of elliptic curve points. The mathematical theory behind elliptic curves and the ECDSA are covered in detail in Chapter 3.

Many standards and implementations make use of digital signatures [11]. Examples are the secure electronic transaction standard (SET) for financial transactions over the internet, the company Identrus for business-related online transactions, the secure sockets layer (SSL) standard for secure communications primarily on the Web and the most commonly used Public Key Infrastructure (PKI) standard ITU X.509 (v3).

The crucial question is whether a digital signature scheme would fit into the resource-restricted framework of RFID applications. While RFID readers can usually afford the

additional computational resources required by such schemes, this is certainly not the case for RFID tags. Intended for mass production at a low manufacturing cost, tags cannot be equipped with complex functionality such as dedicated cryptographic modules. The physical principles on which RFID systems base their operation pose implementation restrictions primarily on the available energy budget. Finally, certain RFID applications, such as Real-time Location Systems (RTLS), require certain performance figures in terms of response delays. These restrictions dictate a careful selection of optimized cryptographic mechanisms that are suitable for use in RFID tags.

A digital signature scheme suitable for RFID applications is considered in this work. Tags are the signature generating devices and readers have the task of verifying the received signatures. The foremost choice is that of the most efficient cryptographic algorithm. While digital signature algorithms, such as RSA, DSA and ECDSA, all provide the desired functionality, the most efficient one for RFID implementations is the one with minimum area and power consumption requirements. An algorithm implementation in general requires more resources for larger inputs. Furthermore, the larger the input, the higher security level the algorithm offers, meaning that higher computational effort is required in order to break the system. Finally, input size affects a number of other aspects of an RFID system, such as radio communication time, key management complexity etc. Here, the *size* is defined to be the number of bits needed to represent the input using a reasonable encoding. In [21] the author concludes that for a given security level, elliptic curve cryptography uses a smaller input. For instance, a 160-bit input for an EC algorithm provides the same level of security as a 1024-bit input for RSA. Regarding power consumption the study in [4] concludes that ECDSA consumes significantly less power when compared to RSA and DSA of comparable security levels, especially for the signature generation part. Signature verification is more efficiently performed by RSA. However, the signature generating devices (RFID tags) impose the most restricting power budget and therefore ECDSA becomes a more attractive choice for implementation.

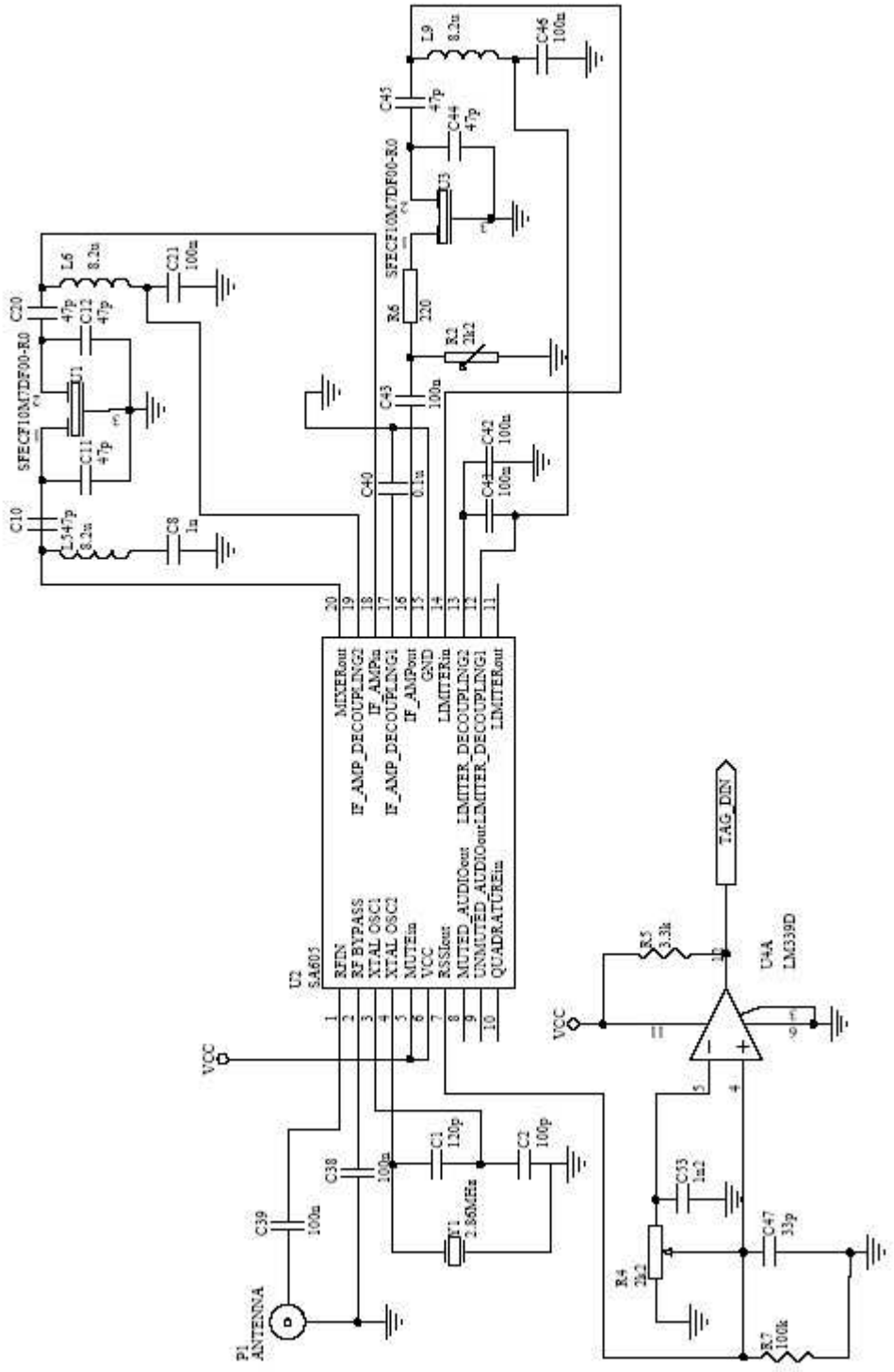

Figure 2.14: Complete Design of Tag Receiver

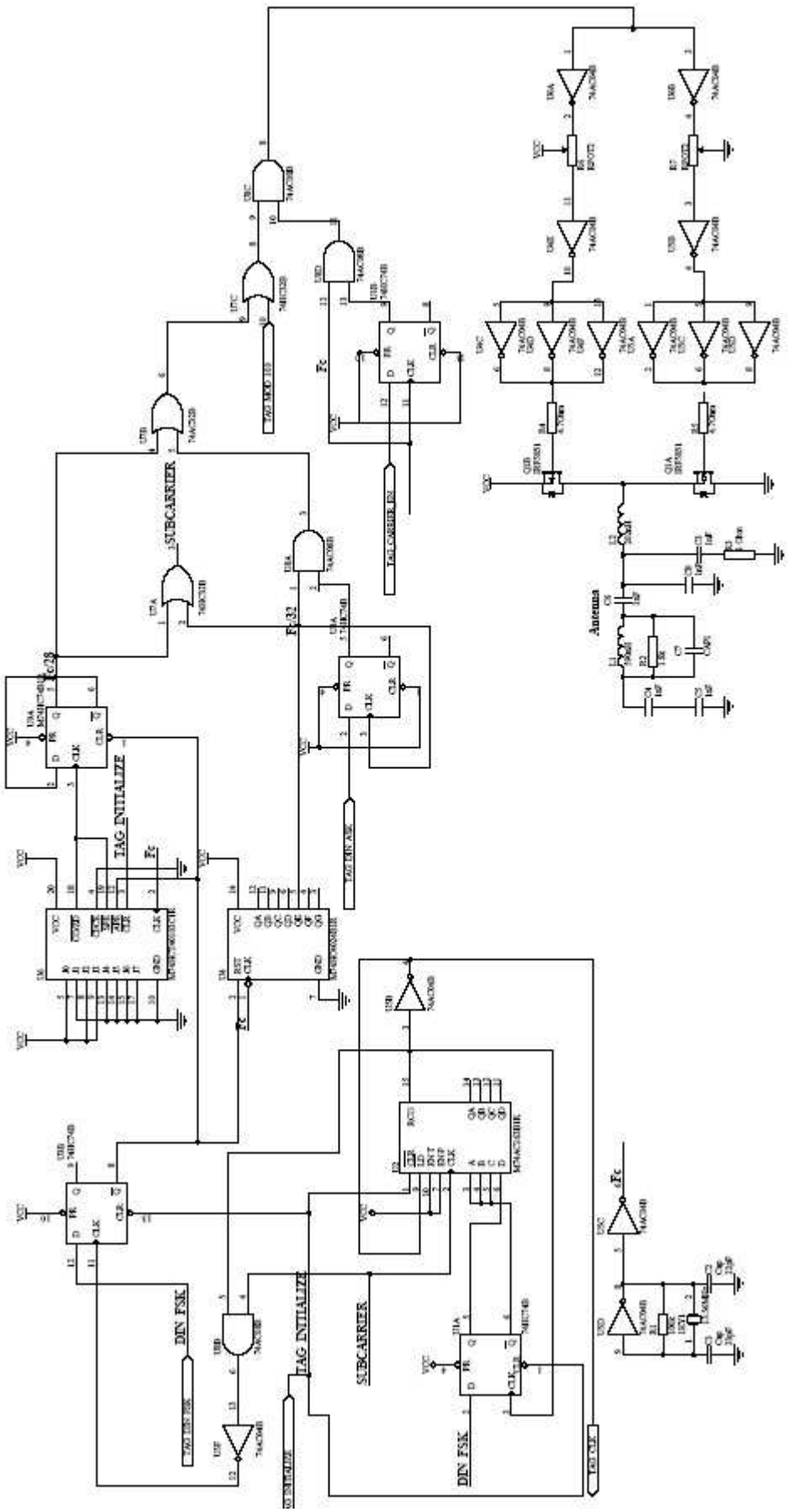

Figure 2.15: Complete Design of Tag Transmitter

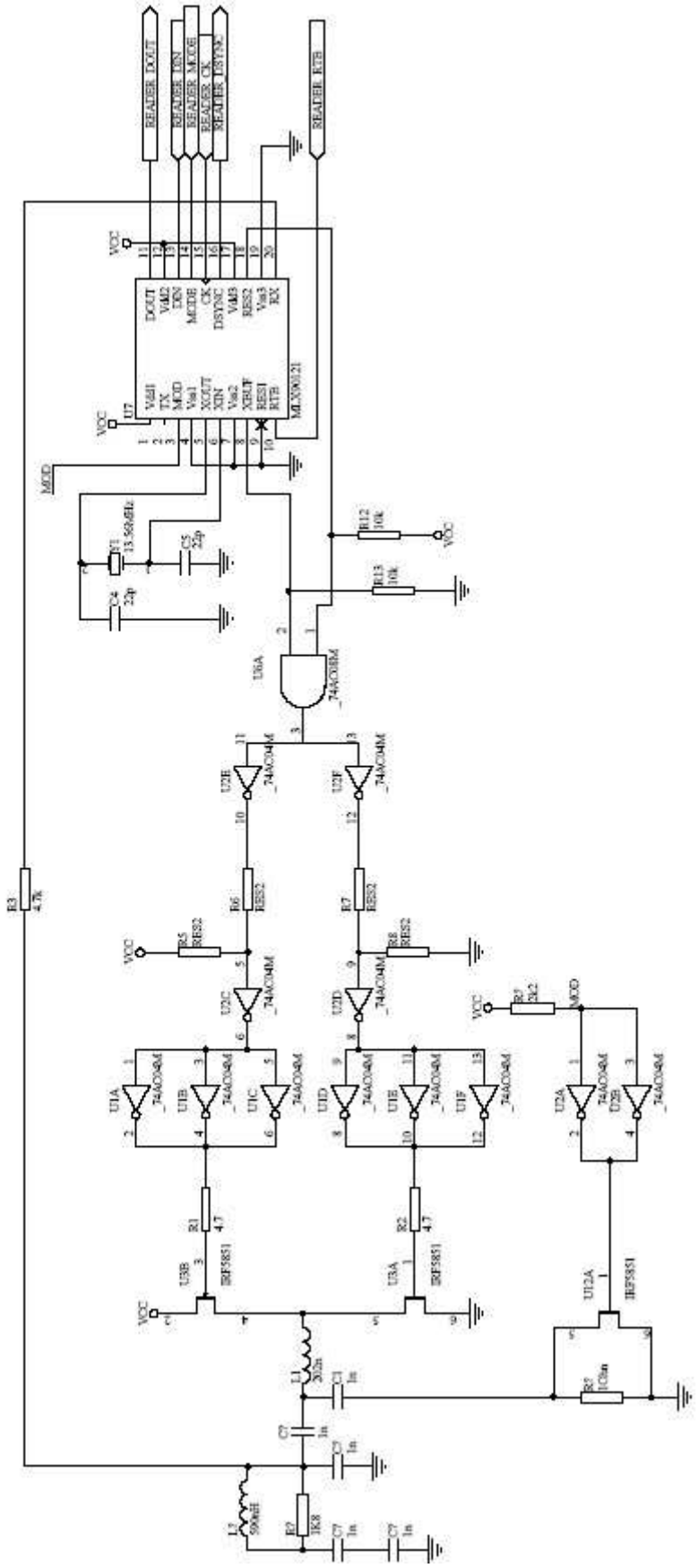

Figure 2.16: Complete Design of Reader Side (transmitter and receiver)

Introduction to Elliptic Curve Digital Signature Generation

3

The theory of elliptic curves has been developed by mathematicians for over a hundred years. Using them as a basis for public-key cryptography was first proposed as late as 1985, independently by Neal Koblitz [29] and Victor Miller [35], with intense research on the topic following subsequently. The suggested public-key cryptosystem would be based on existing ones, such as the DSA, and would take advantage of the properties of elliptic curves to provide sufficient security strength with more efficient implementations. As elliptic curves were gaining on popularity, certain algorithms were standardized and have since attracted considerable commercial attention. One such example is the Elliptic Curve Digital Signature Algorithm (ECDSA).

In this chapter, an introduction to the theory of elliptic curves is attempted, in order to build on the mathematical background required to comprehend the calculations involved in an elliptic curve cryptosystem. The latter uses elliptic curve operations and their corresponding mathematical properties for cryptographic purposes. These operations are performed within the frame of finite fields and their operations, which are also introduced in this chapter. Since the topic itself is mathematically complex and can lead to profound analyses beyond the purpose of this thesis, only the essential concepts are herewith presented, leaving the details to the referred literature. The algorithms presented in this chapter are restricted to the ones that are used in the hardware implementation presented in Chapter 4. Finally, relevant cryptographic algorithms that use elliptic curves are described, with emphasis put on ECDSA.

3.1 Mathematical background

This section presents the basic mathematical concepts and entities in order to describe the elliptic curve theory and the relevant algorithms. Starting from Abelian Groups and Finite Fields, subsequently the principles of arithmetic in prime and binary finite fields and finally elliptic curves and operations on elliptic curves are presented. In this last part, we focus primarily on the most important elliptic curve operation for the subject, the scalar multiplication.

3.1.1 Abelian Groups

An abelian group G (represented by $(G,*)$) is defined by a set G with a binary operation $* : G \times G \rightarrow G$ satisfying the following properties:

- *Associativity*: $a * (b * c) = (a * b) * c, \forall a, b, c \in G$

- *Identity*: there exists an element $e \in G$ such that $a * e = e * a = a, \forall a \in G$
- *Inverse*: $\forall a \in G$ there exists an element $b \in G$ such that $a * b = b * a = e$
- *Commutativity*: $a * b = b * a = a, \forall b \in G$
- *Closure*: For $a, b \in G, c = a * b \in G$.

3.1.2 Finite Fields

Finite fields or Galois fields are abstractions of number systems and their properties. A finite field F_p (or $GF(p)$) comprises a finite set F along with two basic operations, addition (+) and multiplication (*), which satisfy the following properties:

- (F, +) is an abelian group with additive identity represented by 0.
- ($F \backslash [0], *$) [1] is an abelian group with multiplicative identity represented by 1.
- The distributive law holds for $GF(p)$: $(a + b) * c = a * c + b * c, \forall a, b, c \in F$.

Besides addition and multiplication, subtraction and division are also supported and are defined in terms of addition and multiplication respectively. The additive inverse of a field element b is called the negative of b and is represented by $-b$.

The finite field abstraction is necessary for the efficient representation of finite-precision arithmetic that is used in computer hardware, instead of the familiar number sets of real, integer or rational numbers. Some useful definitions on finite fields follow: The *order* of a finite field is the number of elements that are comprised in the field. A finite field of order q exists if and only if q is a power of a prime, $q = p \cdot m$, with the prime p called the characteristic of the field and m a positive integer. If $m = 1$ then F is called a *prime field* and if $m > 2$ it is called an *extension field*. Two fields of the same order are structurally the same, they only differ in the way elements are represented and are therefore called isomorphic. In essence then, for each prime power q there is only one field, denoted by F_q.

A *polynomial* over an arbitrary finite field F is a mathematical expression of the form $f(x) = f_0 + f_1 x + f_2 x^2 + .. + f_n x^n$, where n is a positive integer, called the polynomial *degree*, x is an indeterminate over F and $f_i \in F$ are the polynomial *coefficients*. Polynomials can be added and multiplied and we say that a polynomial $g(x)$ divides a polynomial $f(x)$ if there exists a polynomial $h(x)$ so that $f(x) = g(x)h(x)$. A polynomial $p(x)$ over the finite field F that is only divisible by $ap(x)$ or a, with $a \in F$, is called *irreducible*.

[1] denotes a group excluding zero element

Some types of finite fields are of special interest, for elliptic curve cryptosystems. These are 1) the prime fields mentioned earlier, 2) finite fields with characteristic 2 ($q = 2^m$) called binary fields and 3) optimal extension fields of prime characteristic. The choice of finite field and finite field specifics (e.g. element size) used for computation, greatly affects a hardware implementation. Prime fields, denoted by $GF(p)$, make use of conventional integer arithmetic, combined with a modular reduction step. Field elements are integers [0, .., p-1] and the modulus is a prime, which can be of a size of more than 190 bits for cryptographic applications. Binary extension fields (or simply binary fields) are fields of characteristic two and order 2^m, and are denoted by $GF(2^m)$. Field elements are represented by polynomials with coefficients in $GF(2) = 0,1$. It can be proved that the modular reduction operation in the polynomial case resembles the integer case and can thus define a finite field similar to the prime fields, if an irreducible polynomial is used as the modulus. For cryptographic applications, polynomials of degree $n \geq 160$ are used. The third interesting type of finite fields is the optimal extension field, which is a generalization of the binary extension field for prime characteristic larger than 2. In this case, elements are represented by polynomials with coefficients from small prime fields. Generalized extension fields are not supported by the ECDSA standard.

3.1.2.1 Prime Field Arithmetic

As mentioned above, prime field arithmetic resembles integer arithmetic. This means that all the well-known number representations and arithmetic operation techniques can be used to perform the necessary computations. Choices for number representation are redundant and non-redundant numbers, signed and unsigned, each with its advantages and disadvantages. The decision is made according to the specific design constraints. For instance, the familiar binary representation (a positional number system with radix 2) requires the least storage resources, as opposed to stored carry form (where intermediate results store the carry separately) that supports fast arithmetic circuitry. An elaborate discussion on number representations and their properties can be found in [42]. Negative numbers depend on the number representation of choice. A typical choice is the 2's complement one, that requires simple logic for arithmetic, besides sign and magnitude comparisons which are necessary for diverse operations. In *2's complement binary representation*, numbers are represented as in Equation (3.1):

$$(x_{k-1}, x_{k-2}, .., x_1, x_0), x = -2^{k-1} \cdot x_{k-1} + 2^{k-2} \cdot x_{k-2} + .. + 2 \cdot x_1 + x_0, x_i \in [0, 1] \quad (3.1)$$

Addition in $GF(p)$ is performed modulo p, as the corresponding algorithm shows. Since the prime modulus is not a power of 2, a simple integer adder is not adequate and a reduction step following the addition is necessary. As described earlier, subtraction is also expressed through addition and thus the same algorithm holds.

The modular reduction operation brings the operand to the range [0, p-1], by adding/-subtracting an appropriate integer multiple of the modulus. This integer is called quotient and can be calculated either by successive additions/subtractions of the modulus and subsequent comparisons to the modulus, or by on-the-fly calculation of the quotient and a single addition/subtraction of the modulus. The latter solution is of course much

Algorithm 1: Addition in Prime Fields

Input : $a, b \in GF(p)$, p prime
Output: $(a + b)$ mod p
1 $c \leftarrow a + b$
2 **if** $c \geq p$ **then** return $c - p$
3 **else** return c

more expensive in computational resources. Special forms of the modulus (such as the *generalized Mersenne primes* of the form $p = 2m - p'$, with p' a small-valued integer) assist in simplifying this costly operation. Another technique is using a quotient estimate, instead of the precise value, to perform the reduction with a possible final correction step. This technique is especially useful when the chosen number representation prohibits fast comparisons (as is the case in stored-carry representations) and is presented in [15].

The most important operation for cryptographic algorithms is modular multiplication, $c = (ab)$ mod p, not only for elliptic curve operations but for other algorithms such as the RSA, where modular exponentiation is performed. Significant computation time of such algorithms is spent on multiplication and therefore designers focus on optimizing this first. This is confirmed also by the number of research papers treating this topic, such as [49], [18], [32] etc. Multiplier implementations come in two flavors; *fully parallel multipliers* process whole operands in a single cycle and *digit-serial multipliers* process operands in more cycles, consuming groups of operand bits per cycle in order to generate and accumulate partial products. The digit size varies depending on the computational resources available for the design of the multiplier, the simplest solution being a bit-serial version, processing one multiplier bit at a time. Fully multiple multipliers are prohibitive for cryptographic-size operands and thus multi-cycle units are preferred. Multiplier digits can be processed MSB-first or LSB-first. Additionally, booth recoding techniques can be applied, in order to reduce the number of non-zero bits of the multiplier, and thus the total number of accumulations necessary for multiplication. The modular reduction step can take place either after the complete multiplication operation (requiring though larger sizes for intermediate storage and computational resources) or after each intermediate partial product accumulation. A bit-serial modular multiplication algorithm, with interleaved modular reduction is presented in Algorithm (2).

Modular inversion is another costly operation, that calculates the multiplicative inverse of a field element. There are three principal algorithms for modular inversion, namely one based on the extended Euclidean algorithm, with iterative transformations of the greatest common divisor function, one based on Fermat's Little Theorem, based on modular exponentiation and one based on solving a system of linear equations with Gaussian elimination. Here, the second one is described. The theorem states that a^{p-1} mod $p \equiv 1$ and therefore the inverse can be calculated as

$$a^{-1} = a^{p-2} \, mod \, p \tag{3.2}$$

using Algorithm (3) for exponentiation. Squaring is computed through modular multiplication. For this specific case, a choice of p such that p-2 has only a few non-zero digits in its binary representation helps reducing the number of necessary multiplications.

Algorithm 2: Bit-serial multiplication for prime fields with interleaved modular reduction

Input : $a, b \in [0, p-1], 2^{m-1} < p < 2^m$
Output: $(ab) \bmod p$

```
1 c ← 0
2 for i=m-1 downto 0 do
3     c ← 2c + ab_i
4     c ← c mod p
5 end
6 return c
```

Algorithm 3: Square-and-multiply algorithm for modular exponentiation

Input : $a \in [1, p-1], e = (e_{m-1}, e_{m-2}, .., e_1, e_0)$, prime p
Output: $c = a^e \bmod p$

```
1 c ← 1
2 for i=m-1 downto 0 do
3     c ← c^2 mod p
4     if e_i = 1 then
5         c ← (ca) mod p
6     end
7 end
8 return c
```

One particularly helpful technique, for efficient implementation of modular arithmetic, is arithmetic in the Montgomery domain (named after Peter L. Montgomery who introduced it in 1985[36]). In essence, integers are converted into fractional numbers so that arithmetic becomes easier and modular reduction becomes implicit. Furthermore, all arithmetic operations are supported in the Montgomery domain, and are more efficient (especially multiplication, which was the initial improvement target when this technique was proposed). It is necessary, though, to convert the result back to the integer domain. The total conversion overhead justifies utilization of this technique only when a large number of operations is to be performed in the Montgomery domain and not for a single multiplication.

A more profound treatment of prime finite fields and arithmetic performed in these fields can be found in [21]. The same book includes a detailed treatment of binary field arithmetic, an overview of which follows in the following subsection.

3.1.2.2 Binary Field Arithmetic

Arithmetic in binary fields permits significantly more efficient hardware implementations. The essential difference when compared to prime field arithmetic, is the absence of carry propagation in addition operations. As a result, much faster or much simpler circuitry can be utilized in favor of circuit performance, encompassing the full range of arithmetic

operations.

A binary extension field $GF(2^m)$ of $GF(2)$ can be viewed as a vector space of dimension m over $GF(2)$. Vectors are elements of $GF(2^m)$ and scalars are elements of $GF(2) = 0,1$. If $\{b_1, b_2, .., b_m\}$ is a basis of $GF(2^m)$ over $GF(2)$, then an arbitrary element $a \in GF(2^m)$ can be represented as a linear combination of the basis elements, or $a = a_1b_1 + a_2b_2 + .. + a_mb_m$. In general there are many distinct bases that can be used, but only three types are of particular interest. The first type is of the form $\{1, a, a_2, .., a_m\}$, where a is a root of the irreducible polynomial $p(x)$ of degree m, used to define the field over $GF(2_m)$ and is called *polynomial basis*. Elements are represented as polynomials of degree at most m-1, whose coefficients are defined over $GF(2) = 0,1$. The second type is of the form $\{a, a^2, .., a^m\}$, or the set of conjugates of a suitable element of $GF(2^m)$. A convenient choice is the set of roots of a prime polynomial $p(x)$ if it is certain that they are linearly independent. This type is called *Gaussian Normal Basis*. A third basis is the dual basis, which receives however little attention in the literature and therefore will not be mentioned any further. Elliptic curve cryptography standards propose specific finite fields, both in polynomial and normal basis representation. According to [32], however, normal basis representations have no particular advantage over polynomial basis ones and therefore only the latter case will be considered in this thesis. Elements are represented as a bit string of length m, comprising the binary coefficients of the corresponding polynomial, in polynomial basis representation. For polynomial bases, the irreducible polynomial that defines the field is usually given in the form of a trinomial ($p(x) = x^m + x^k + 1$) or a pentanomial ($p(x) = x^m + x_1^k + x_2^k + x_3^k + 1$) in order to obtain efficient implementations.

As explained previously, addition in binary field arithmetic is extremely simple as it is a carry-free operation. Addition takes place coefficientwise, and since coefficients are elements of $GF(2)$, a simple XOR operation suffices. Furthermore, the result is automatically reduced and therefore no explicit reduction step is required. Finally, subtraction is the same operation as addition, as the additive inverse of a field element is the element itself (as $x \oplus x = 0$).

Modular reduction can also be very efficient in $GF(2^m)$. The operation in principle is the same as in the prime field case for an arbitrary modulus. However, when a particular polynomial is considered as a modulus, field properties enable some mathematical shortcuts. In specific, the polynomial can be written as $p(x) = x^m + p'(x)$. Since $p(x) = 0 \mod p(x)$, the following congruence holds:

$$x^m = p'(x) \, mod \, p(x) \tag{3.3}$$

Using Equation (3.3), each power $x^{m+i}, i \geq 0$ in the polynomial to be reduced, can be substituted by $p'(x)x^i$ without affecting the congruence modulo $p(x)$. If the degree of $p'(x)$ is chosen to be much smaller than m, then the degree of the operand is effectively reduced. Iteratively applying this transformation, results in a fully reduced field element. This technique allows simple logic to perform modular reduction in a single step. Similar methods exist also for the prime field case, since an equation similar to Equation (3.3) holds. If the prime modulus be expressed as a sum of a small number of terms, which is typically the case in cryptographic standards, also prime field reduction can benefit from this property, although more complex logic is involved.

With addition and reduction greatly simplified, modular multiplication will be more efficient than the corresponding prime field version. Multiplication algorithms can also be digit- or bit-serial, or in the Montgomery domain (although the latter case is not as beneficial as over prime fields). For bit-serial algorithms, two approaches exist, the *MSB-first* and the *LSB-first*, with comparable resource requirements and performance in terms of speed. The former case is presented in Algorithm (4). There are obvious similarities with the prime field counterpart, as expected, with $c(x)x$ representing a shift-left operation.

Algorithm 4: Bit-serial multiplication for binary fields with interleaved modular reduction

Input : $a(x), b(x) \in GF(2^m)$, $p(x)$ irreducible polynomial of degree m
Output: $(a(x)b(x))$ mod $p(x)$

```
1 c(x) ← 0
2 for i= m-1 downto 0 do
3     c(x) ← c(x)x + a(x)b_i
4     c(x) ← c(x) mod p(x)
5 end
6 return c(x)
```

Another operation that can be performed efficiently in binary fields, is squaring. When squared, operand $a(x)$ can be written as

$$\begin{aligned} a^2(x) &= a(x)a(x)\, mod\, p(x) \\ &= (a_h(x)x^i + a_l(x))^2 \\ &= (a_h(x)x^i)^2 + a_h(x)x^i a_l(x) + a_h(x)x^i a_l(x) + (a_l(x))^2 \\ &= (a_h(x)x^i)^2 + (a_l(x))^2. \end{aligned}$$

Recursive application of this transformation leads to individual squaring of each term of the polynomial or equivalently

$$a^2(x)\, mod\, p(x) = \sum_{i=0}^{m-1} a_i x^{2i} = (a^{m-1}, 0, a^{m-2}, 0, .., a^1, 0, a^0). \tag{3.4}$$

Subsequently, a modular reduction step brings the result to the desired range.

Inversion in binary fields is based on the same algorithms as in the prime field case. Here, Algorithm (5) is based on the binary version of extended Euclidean algorithm. Let the input operand be $a(x)$. Since $p(x)$ is irreducible, the following equality holds: $gcd(a(x), p(x)) = 1$, with function $gcd()$ calculating the greatest common divisor of the two operands. It is known that the greatest common divisor can be written as a linear combination of the two operands, i.e. $a(x)s(x) + p(x)r(x) = 1$. This equation leads to $a(x)s(x) = 1$ mod $p(x)$ and therefore $s(x)$ will be the multiplicative inverse of $a(x)$. The algorithm then iteratively transforms this relationship, keeping it invariant throughout the variable updates, until the inverse is calculated. In the pseudocode

given in Algorithm (5), the notation y_0 denotes the least significant bit in the binary representation of y. The algorithm takes three properties into account:

- if both polynomials $a(x)$ and $b(x)$ are even (i.e. the least significant bit is zero) then $gcd(a(x), b(x)) = gcd(a(x)/2, b(x)/2)$
- if $a(x)$ is even and $b(x)$ is odd or vice versa, $gcd(a(x), b(x)) = gcd(a(x)/2, b(x))$, and
- if both polynomials are odd, $gcd(a(x), b(x)) = gcd(a(x) - b(x), b(x))$.

It should be noted that if $b(x)$ is initialized with the polynomial $n(x) \in GF(2^m)$, modular division $(n(x)/a(x) \bmod p(x))$ can be obtained instead of simple inversion, hence incorporating one multiplication in the operation and saving computation time. An inversion algorithm for prime field arithmetic is also based on the same principles.

The major hindrance against using division, traditionally has been the long computational time it requires. In specific, it takes $2m$-1 iterations to complete the inversion. To make matters worse, the comparison operation can be very inefficient, especially when cryptographic-length operands are involved. A remedy for the latter problem was firstly proposed in [14], where a counter was used to replace the costly comparison. This improvement permitted digit-serial algorithms to be designed (as opposed to bit-serial) in order to reduce the required number of iterations, to complete the algorithm. One such proposed algorithm can be found in [17]. These approaches help in making modular division a more attractive choice for implementation.

3.1.3 Elliptic Curves

An elliptic curve E over a finite field K is defined by the equation

$$E : y^2 + a_1xy + a_3y = x^3 + a_2x^2 + a_4x + a_6, a_1, a_2, a_3, a_4, a_6 \in K \tag{3.5}$$

This form of the elliptic curve equation is called a Weierstrass equation. Especially for prime and binary extension fields, the elliptic curve equation can be re-written as

$$EC(GF(p)) : y^2 = x^3 + ax + b, a, b \in GF(p) \tag{3.6}$$

in the prime field case and

$$EC(GF(2^m)) : y^2 + xy = x^3 + ax^2 + b, a, b \in GF(2^m) \tag{3.7}$$

in the binary field case. A point on the elliptic curve is a pair of coordinates (x,y) that satisfy the above equation. Point coordinates are elements of the underlying field as well. The set of elliptic curve points, together with point addition form an abelian group and therefore have properties analogous to modular integer arithmetic.

The fundamental elliptic curve operation is addition of two points, which results in a third point on the elliptic curve (due to the closure property of the definition of abelian groups). When a point is added to itself, the operation is referred to as point doubling.

Algorithm 5: Inversion in binary fields based on the binary extended Euclidean algorithm.

Input : $a(x) \in GF(2^m)$, $p(x)$ irreducible polynomial of degree m
Output: $(a^{-1}(x))$ mod $p(x)$

```
1  y(x) ← a(x)
2  d(x) ← p(x)
3  b(x) ← 1
4  z(x) ← 0
5  while y(x) ≠ 0 do
6      while y_0 = 0 do
7          y(x) ← y(x)/x, b(x) ← (b(x) + p(x)b_0)/x
8      end
9      while d_0 = 0 do
10         d(x) ← d(x)/x, z(x) ← (z(x) + p(x)z_0)/x
11     end
12     if y(x) ≥ d(x) then
13         y(x) ← y(x) + d(x), b(x) ← b(x) + z(x)
14     else
15         y(x) ← d(x) + y(x), z(x) ← z(x) + b(x)
16     end
17 end
18 return c(x)
```

A series of point additions is called multiplication of the point with a scalar value (equal to the number of additions), or concisely scalar multiplication. The latter is the core operation in the ECDSA protocol and similar cryptographic algorithms.

Elliptic curve operations are performed as a sequence of finite field arithmetic operations, which are completely dependent on the underlying finite field, as well as the way elliptic curve points are represented. By definition, *affine coordinate representation* implies that only two coordinates are required in order to describe a point on the elliptic curve, which belongs to the two dimensional space. There are, however, other possibilities on representing a point that could be advantageous for a hardware or software implementation. They are collectively termed *projective coordinates* and in principle use a third coordinate z to represent a point on the elliptic curve. A projective coordinate system is derived from the affine representation by means of a change of variables

$$(x, y) \rightarrow (x/z^c, y/z^d) \tag{3.8}$$

Integers c and d determine which projective coordinate system is derived, using the change of variables given in Equation (3.8). For example, for c=d=1, the standard-projective representation is obtained. This substitution effectively allows the denominator of a modular division to be stored in z, which means that this expensive operation can temporarily be avoided. In order to acquire the final result, after all elliptic curve operations have been performed, the point needs to be transformed back to the affine

representation. As a result, only one division will be necessary after the computations on the elliptic curve have been completed. This benefit, however, comes at the cost of an increased number of field operations.

Formulas for the aforementioned elliptic curve operations can be derived for each representation system. Starting from the most basic ones, the point addition formula for affine coordinates is derived from geometrical methods. The resulting formulas, for an elliptic curve defined over a binary field are

$$P_1 + P_2 = (x_1, y_1) + (x_2, y_2) = (x_3, y_3) = P_3 \tag{3.9}$$
$$x_3 = \lambda^2 + \lambda + x_1 + x_2 + a, \tag{3.10}$$
$$y_3 = (x_1 + x_3) + x_3 + y_1, where \tag{3.11}$$
$$\lambda = (y_1 + y_2)/(x_1 + x_2), if P_1 \neq P_2 \tag{3.12}$$
$$\lambda = y_1/x_1 + x_1, if P_1 = P_2 \tag{3.13}$$

Similar formulas hold when prime fields are used. For projective coordinates, the corresponding formulas can be derived from the change of variables mentioned above. An illustrative example is the point addition using the so-called mixed Jacobian-projective coordinates (substitution with c=2, d=3):

$$P_1 + P_2 = (x_1, y_1, z_1) + (x_2, y_2, 1) = (x_3, y_3, z_3) = P_3 \tag{3.14}$$
$$x_3 = (y_2 z_1^3 - y_1)^2 - (x_2 z_1^2 - x_1)^2 (x_1 + x_2 z_1^2) \tag{3.15}$$
$$y_3 = (y_2 z_1^3 - y1)(x_1 (x_2 z_1^2 - x_1)^2 - x_3) - y_1 (x_2 z_1^2 - x_1)^3 \tag{3.16}$$
$$z_3 = (x_2 z_1^2 - x_1) z_1 \tag{3.17}$$

It can be noticed that P_2 is defined in affine coordinates ($z_2 = 1$), which could be beneficial when a series of points in affine representation need to be accumulated. Another observation is that division is no longer required. However, the formulas that calculate the new point's coordinates are noticeably more complicated. In the specific cases, a straightforward implementation would require one inversion and two multiplications for the affine point representation (squaring operations and additions are of trivial cost in binary field arithmetic) and 15 multiplications in the jacobian-projective representation. Furthermore, additional storage space is required for the third coordinate. Consideration of other projective coordinate systems yields similar results. The decision on which coordinate system would be more efficient to use, depends mainly on the relative cost of an inversion over a multiplication. For example, if the cost of an inversion is larger than 10 multiplications, then the Jacobian-projective system of Equation (3.14) should be preferred.

The most important elliptic curve operation is, especially for ECDSA, the scalar multiplication. If k is a positive integer, then kP denotes the point obtained by adding together k copies of the point P. For cryptographic purposes, the scalar k is a large integer. This renders a naive implementation of the scalar multiplication formula, i.e. with k repeated additions of P, prohibitive. Scalar multiplication is by far the most time-consuming elliptic curve operation used in cryptographic schemes and it is certain that it attracts the main focus of relevant scientific research. In all choices of coordinates, point

doubling is less costly than point addition and therefore doubling is preferred to addition whenever this is possible. A simple solution is therefore the *double-and-add* algorithm (6) where m point doubling operations are performed and the number of additions depends on the number of non-zero bits in the binary representation of k (on average $m/2$), m being the size of the binary representation of k ($k \geq 160$). An improvement on this method requires recoding the scalar k in order to minimize the non-zero bits of its representation (termed *Non-Adjacent Form* or NAF) and therefore minimizing the number of point additions. For this purpose, a signed digit representation (i.e. with digits from the set {-1,0,1} is used, so as to obtain a number that has no consecutive nonzero digits. The implementation of such a conversion is however not trivial and this constitutes a drawback for this method. If extra memory is available, it is possible to pre-compute some multiples of P and use them to process more than one bits of the scalar k at the same time, with the so-called window methods. For instance, in order to process w=2 bits of k at a time, multiples P, $2P$ and $3P$ are required. This method is particularly useful when the base point P is fixed, as the multiple pre-computation is a one time operation. Although they offer significant improvements in terms of speed, these methods become very greedy in storage resources, thus occupying considerable chip area.

Algorithm 6: Scalar Multiplication using the double-and-add method

Input : scalar $k = \sum_{i=0}^{m-1}$, Elliptic Curve point P
Output: $Q = k \cdot P$

$Q \leftarrow \infty$
for i = *m-1 downto 0* **do**
    $Q \leftarrow 2 \cdot Q$
    if k_i = *1* **then**
        $Q \leftarrow Q + P$
    end
end
return Q

A very attractive method, popular in relevant designs, is the Montgomery method (also known as Montgomery's ladder). The main idea behind this method is that the x-coordinate alone, of the elliptic curve points under addition, is sufficient in order to obtain the x-coordinate of the resulting point, when the difference of the involved points is known. Consequently, the complete scalar multiplication will be performed on the x-coordinates only, retrieving the y-coordinate by the known difference simply as a last step. This way, the cost in finite field operations is effectively halved. The general algorithm is presented below. It maintains a constant difference of P between points P_1 and P_2 by executing the same operations in each iteration, which are one point doubling and one point addition. Depending on the value of k_i, one point is doubled and the other becomes the result of the point addition. This method was proposed for use with elliptic curves over binary fields in [31] and over prime fields in [26], using both affine and projective coordinates. Algorithm (7) presents an improved version of the original

Montgomery's ladder, for affine representation of elliptic curve points, as seen in [31]. The algorithm requires

1. $2\lfloor log_2 k \rfloor + 1$ divisions;
2. 3 multiplications;
3. $4\lfloor log_2 k \rfloor + 6$ additions;
4. $2\lfloor log_2 k \rfloor + 2$ squaring operations

to complete one scalar multiplication.

Algorithm 7: Improved scalar multiplication algorithm using affine coordinates over GF(2^m)

Input : Integer $k = (k_{l-1}k_{l-2}..k_1k_0) \geq 0$, point $P = (x, y) \in GF(2_m)$
Output: Point $Q = kP$

if $k = 0$ *or* $x = 0$ **then**
  return (0,0)
end
$x_1 \leftarrow x, x_2 \leftarrow x^2 + b/x^2$
for i= *l-2 downto 0* **do**
  $t \leftarrow \frac{x_1}{x_1+x_2}$
  if $k_i = 1$ **then**
    $x_1 \leftarrow x + t + t^2, x_2 \leftarrow x_2^2 + b/x_2^2$
  else
    $x_2 \leftarrow x + t + t^2, x_1 \leftarrow x_1^2 + b/x_1^2$
  end
  $r_1 \leftarrow x_1 + x, r_2 \leftarrow x_2 + x$
  $y_1 \leftarrow r_1(r_1r_2 + x^2 + y)/x$
  return $Q(x_1, y_1)$
end

3.2 Elliptic Curve Cryptography

Scalar multiplication offers a computationally intractable problem, highly suitable for cryptographic purposes. In specific, the elliptic curve discrete logarithm problem (ECDLP) states that by mere knowledge of the points Q and P, and the curve parameters, the scalar k cannot be obtained in less than exponential time, by use of any algorithm known so far. This statement assumes that the elliptic curve and the scalar have been carefully chosen in order to resist all known attacks on ECDLP. To point out the hardness of ECDLP, a team of mathematicians from Texas Tech University, required 2600 computers and 17 months of computation to solve one such problem with a size of k of 109 bits, for the binary field case [16]. There is no formal proof that the ECDLP

problem is actually intractable, but is regarded as such, since there is no sub-exponential time algorithm to solve it. Consequently, systems using cryptographic key sizes of 160 bits or more, are considered to be safe for the coming decade.

The assumed hardness of ECDLP allowed a number of cryptographic primitives and protocols to be created, with more following in the future. These include *digital signature schemes*, *key establishment schemes* and *encryption schemes*. Digital Signatures have been presented in Section 2.4. Key establishment schemes are used to provide two or more parties with a shared secret key, over an insecure network. This key can afterwards be used in a symmetric-key protocol. Key agreement schemes are special cases of key establishment schemes, where all participating parties contribute information in order to derive the shared key. Such schemes, using elliptic curve arithmetic are the Station-to-Station (STS) protocol and ECMQV. Encryption schemes using elliptic curves are not very popular, due to the relatively large computational effort required to encrypt bulk data. For small pieces of data, the most widely known scheme is the Elliptic Curve Integrated Encryption Scheme (ECIES).

An already existing protocol, which has been commercially accepted after becoming a standard by many international standardization bodies, is the *Elliptic Curve Digital Signature Algorithm* [40], [38]. The ECDSA is used by a signatory to generate a digital signature on data and by a verifier to validate the authenticity of the signature. The algorithm fits in the context of asymmetric cryptographic techniques, where each signatory uses a private key for the signature generation process and the verifier uses a public key for the signature verification process. Algorithm (8) depicts the pseudocode as described in the standards. Private key is a randomly generated integer d and the public key Q is generated by scalar multiplication of another randomly generated integer k with a base point P, or $Q = kP$. In fact, this is the only computation in the standard that takes place purely on elliptic curves and the underlying finite field. The x coordinate of Q is then converted to an integer and the subsequent computations in Algorithm (8), which lead to the signature, are performed modulo a large integer (the order of P), therefore the underlying field is considered to be $GF(p)$. As a result, computations for signature generation may be performed either completely over a prime field or over a binary extension field (scalar multiplications) and a prime field (rest of computation). Since binary field arithmetic is more efficient, designers usually select to support the latter choice for their implementations. Signature verification is a more computationally intensive operation, just because it requires two scalar multiplications, instead of one, as can be seen in Algorithm (9).

ECDSA, as well as all other digital signature algorithms in use, do not alter the message to be transmitted M and append the generated signature to the message. The signature depends on both the private key of the user and the message M, which is first compressed in order to reduce the size of the data to be processed. Message compression takes place with the aid of a cryptographically secure hash function $H(M)$ (SHA-1 [3] is recommended in the standards) and the resulting data is referred to as the message digest. The hash function must make sure that it is very difficult for two different messages to hash to the same value, and for the initial message to be retrieved by knowledge of the hash value. In domain terminology, the hash function must be collision resistant and pre-image resistant. The message digest also needs to be computed again for signature

verification.

Another operation in ECDSA is the random or pseudorandom generation of the integer k, also known as the *nonce*. This nonce is used only once per signed message and subsequently is discarded. If the nonce is generated in a pseudorandom way, the random number generator must ensure that the result is unpredictable. The corresponding standards recommend such generators, based on the SHA-1 hash function or the Data encryption standard (DES). However, other FIPS approved generators can be used instead. A seed is either supplied internally by the user, externally by the system or as a combination of the above.

A series of computations can be performed without knowledge of the message M itself, such as the scalar multiplication, the nonce inversion and the product $d \cdot r$. This way, the necessary computations can be allocated in a more efficient way in time and signature generation can be sped up significantly, when the timely response of the signatory is important.

Algorithm 8: Elliptic curve digital signature generation

Input : Elliptic curve E, base point $P \in E$, $n =\mid P \mid$, private key d, message M
Output: Signature (r, s)

1 k=random(1,n-1)
2 R=$(x_R, y_R) = kP$
3 $r = x_R \bmod n$
4 **if** r=*0* **then** goto line 1
5 $e = H(M)$
6 $s = k^{-1}(e + dr) \bmod n$
7 **if** s=*0* **then** goto line 1
8 return (r, s)

Algorithm 9: Elliptic curve digital signature verification

Input : Elliptic curve E, base point $P \in E$, $n =\mid P \mid$, public key $Q \in E$, message M, signature (r, s)
Output: Signature Valid/Invalid

1 **if** $r \notin (0, n)$ *or* $s \notin (0, n)$ **then** return *Invalid*
2 $e = H(M)$
3 $w = s^{-1} \bmod n$
4 $u_1 = (ew) \bmod n$, $u_2 = (rw) \bmod n$
5 R=$(x_R, y_R) = u_1 P + u_2 Q$
6 **if** R=∞ **then** return *Invalid*
7 **if** x_R=r *mod* n **then** return *Valid*
8 **else** return *Invalid*

The scalar multiplication operation is performed on a particular elliptic curve, which is determined by the choice of the underlying finite field and the curve parameters a and b, which are elements of the chosen field. The finite field is defined as prime or binary,

along with the prime modulus or irreducible polynomial, respectively. Another required parameter is the base point P, which is used also by the receiver for the signature verification process. Necessary parameters are the order n of P and the cofactor $h = | EC(GF(q)) |/n$ (which is usually a small-valued integer, e.g. 1, 2 or 4 for the curves specified in [38]) as well. Additionally, the seed S used for generating elliptic curve parameters is supplied.

By using the pseudorandom number generation techniques described in the standard, an entity can generate the necessary system parameters autonomously. However, certain tests need to be performed in order to ensure that the generated parameters do not correspond to a weak (security-wise) elliptic curve, or otherwise the system might be compromised. One example is the base-point order n, which needs to be sufficiently large, for security demands. The scalar $k \in [0, n-1]$ will be easier to find (even with a straightforward brute force attack) if the order is too small. The latter is hence chosen to be as large as practically possible (e.g. in [38] the smallest acceptable orders are integers with a $\sim$ 192-bit wide binary representation). Testing the point order can be a very computationally intensive attempt. Due to this test, as well as other similar tests, it is often chosen to avoid generating these parameters at random, in resource-constrained implementations like RFID tags. A more powerful computer is used instead, to calculate the parameters and perform the tests. Subsequently, the parameters can be transmitted, in a trusted way, over to the entity, or even be stored permanently in it, as a one time operation. Using a fixed elliptic curve is allowed by the standards and is considered sufficiently secure. It is therefore an attractive option for small embedded devices with limited resources.

3.3 ECDSA Security Considerations

Although ECDLP is assumed to be a hard computational problem, this fact alone does not guarantee that an elliptic curve cryptographic primitive will be adequately secure. The integrity of the system depends upon the prevention of unauthorized disclosure, modification or substitution of the private key d, the nonce k and the seeds for the random number generator. This also holds for the parameters discussed above. In order to prevent compromising the system in one of these ways, the designer needs to pay special attention to the design. As far as the choice of elliptic curve is concerned, some special classes of curves, including the so-called supersingular ones, have been prohibited from standards (such as [7]) because there are known methods for simplifying the ECDLP problem for those cases. Further restrictions on the selection of the elliptic curves can be found in the document itself.

Since solving computationally hard mathematical problems is an active topic within the scientific community, users of ECDSA are advised to keep up with the state of the art in this area, as in case a better algorithm for solving ECDLP is found, then the system's security cannot be guaranteed. For instance, the X9.62 standard is based on the state of the art of 1998. In case significant progress is made on the algorithms solving ECDLP, the standard shall seize to be valid. The size of the key used is also a matter of concern. According to current standards, the key should be longer than 150 bits for short-term security, and longer than 180 bits for medium-term security.

ECDSA standards recommend usage of the SHA-1 secure hash function [3] in order to perform some of the required operations. However, it is stated that if SHA-1 is broken, then the standard should be updated. Unfortunately, SHA-1 is now considered to be broken, according to the work in [51], which showed that the necessary number of operations in order to provoke a collision in the hash function can be significantly reduced. Although a collision is still not trivial to achieve, stronger hash functions, belonging to the SHA-2 group ([3]) of hash functions are currently being considered and will be used in the next version of the standard. One approach that makes attacks more difficult to succeed is if the design does not allow direct external control of the hash function. Consequently, a brute force attack is more difficult to attempt, without physical tampering with the system.

One important advantage of the Montgomery's ladder method for scalar multiplication, which was presented above, is that it is resistant against a number of side-channel attacks. Side-channel attacks are passive attacks on a system, which do not attempt to tamper with it but rather exploit information that is not directly related to the operation. Examples are the execution time of operations, the power consumption of the device during computation, the emission of electro-magnetic radiation during computation, acoustic noise, error messages etc. Without delving into the intricate details of each case, a couple of indicative example cases are presented. Timing attacks exploit the varying latency of cryptographic algorithms. In the case of the double-and-add algorithm, the execution time of one iteration, depends on the value of the bit of the scalar k being processed. On the contrary, the Montgomery method performs exactly the same operations in each iteration regardless of the value of k. This fact also proves important in the case of so-called simple power attacks. The latter exploit the varying power consumption, depending on the specific operation being performed at a certain moment. If, for instance, point addition consumes more power than point duplication, it is fairly easy to retrieve the scalar k when the double-and-add method is used. This is not the case for the Montgomery method.

The use of changing keying material (e.g. the per message nonce k) also helps in securing an ECDSA implementation. Also changing the elliptic curve parameters, wherever this is a viable solution strengthens the design. Another approach for secure implementations is to design a tamper-resistant entity. By tampering attacks, a number of attacks is denoted, that aims at getting physical access to the memories of the entity. These attacks come from well-funded adversaries, such as governmental agencies or large companies, who can afford the necessary equipment. There are no elliptic-curve specific methods to counter this type of attacks. Tamper-resistant implementations use secure memories that erase their contents once a tampering attempt is detected. Other simple counter measures include implementing completely autonomous units, that will not store intermediate or resulting data in external memories, using special storage techniques with redundant bits that require specific masks in order to read comprehensible data and more.

4 EllipticCore Implementation

In this chapter, a module that computes digital signatures using the ECDSA algorithm is presented. The module (named *EllipticCore*) is designed to be used in RFID applications, where resources are constrained by physical and economical factors. The mathematical background of the underlying operations was presented in Chapter 3. In this chapter, the design requirements are analyzed, the design choices in order to meet the requirements are explained and the final design is presented in the following paragraphs. The design was synthesized using 0.35 μm CMOS technology and the synthesis results are finally presented in this chapter, for the purpose of comparing them with previous work.

4.1 Design requirements

Beyond meeting the functional specifications, the goal of digital system design is to optimize one or more quality metrics, depending on the application requirements. These include speed of calculation, occupied area on an integrated circuit (related to manufacturing costs of the IC) and power consumption during operation. Furthermore, the robustness of the design against manufacturing variations or environmental noise, as well as the resistance against security threats are issues that need to be considered in real-world designs. For a thorough discussion on the quality metrics used to evaluate a digital design, see [27]. Optimization on one figure of merit generally counteracts the others and in most cases a balance among the three is desirable. System design can be then interpreted as judicious selecting of computing resources, in order to achieve acceptable results in all aspects considered. In the case of an RFID tag, the performance requirements are set by both physical and economical constraints, although such requirements have not been formally defined yet. Estimations concerning these constraints are presented in the following paragraph.

4.1.1 Power, Area and performance requirements

A short battery life of RFID tags hinders massive-scale deployment or placement at spots that are difficult to reach, as battery replacement will become difficult. In the case of RFID tags, which are powered by the electromagnetic field generated by RFID readers, power consumption affects the maximum operating range of the device. In order to provide more energy, a stronger magnetic field is required, or the tag needs to be brought closer to the reader. This could have negative consequences on a variety of applications. In this case the instantaneous power must be controlled, making the design of such a system even more challenging. A limit of $\sim 50\mu W$ is mentioned in most relevant designs [12, 53, 30].

Increased chip area leads to increased manufacturing costs for the device, and one of

the attributes advertised by RFID system manufacturers is the very low cost per tag. Security enhanced tags are naturally expected to have higher cost. Related designs for elliptic curve cryptography on RFID tags, such as [30], have a gate count in the ~ 20000 ballpark, or area of 1-2 mm^2.

Performance in terms of latency will also be considered in cases where a real time RFID application is involved and the response delay of the tag is critical. The system latency should nevertheless be kept so low as not to become noticeable by the user, which should be no more than a few milliseconds. If direct integration of the system into an existing RFID protocol such as ISO 15693 is desired, then the specific timing constraints should also be considered (in Section 2.1.2 these are specified to be in the range of $[318.8, 323.3](\mu sec)$ for ISO 15693).

4.2 CMOS Power consumption considerations

Power consumption is increasingly attracting designers' attention. Even though high performance designs did not initially focus on it, recent devices' power density has reached such high levels that it has inescapably become an important issue (e.g. overheating problems are caused). The primary application area, however, where power reduction techniques are particularly desirable, is portable devices. Keeping average power consumption (or total consumed energy) at low levels is important when battery-operated devices are concerned, so as to prolong battery life as much as possible. In cases such as passive RFID tags, where no battery is available and the power constraints are even more stringent, power consumption becomes a factor of critical importance. It is therefore worthwhile to look deeper into the factors that cause power dissipation in digital circuits and the techniques typically used to reduce it.

Power dissipation in a digital IC comprises a *static power* and a *dynamic power* component. The static component is caused by leakage currents due to the physical properties of transistors and digital logic circuits implemented on semiconductors. In practice, logic gates exhibit static power consumption that is several orders of magnitude lower than the dynamic counterpart and therefore static power can be considered negligible in most of the cases. An exception regards circuits with extremely low operating frequencies, where static power consumption can be of particular interest.

The dynamic component is due to the transient switching behavior of digital circuits. A formula that gives the dynamic power consumption of a CMOS gate is:

$$P_{dynamic} = \alpha_1 C_L V_{DD}^2 f_{clk} \tag{4.1}$$

Consider a CMOS inverter, such as the one in Figure 4.1, powered by a supply voltage V_{DD}. The gate drives a capacitive node with capacitance C_L, which reflects the input capacitance of gates connected to this node and the distributed capacitance on the interconnecting wire. When the input voltage switches logical state, the load capacitance is charged or discharged. In the example of Figure 4.1, V_{in} begins in logical high state, and according to the theory the PMOS transistor turns on and the NMOS transistor is switched off. This means that the load capacitance is charged so that node V_o has a voltage of V_{DD} with respect to the ground. When input voltage switches to logical

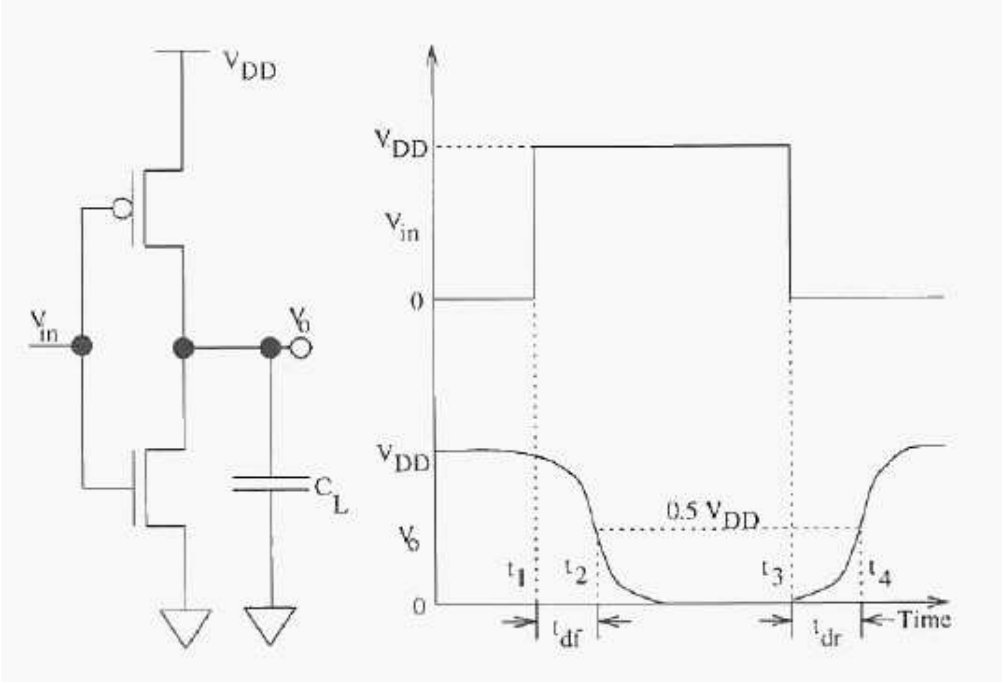

Figure 4.1: A CMOS inverter and the voltage waveforms while switching

low, the gate switches states, i.e. the PMOS transistor is off and the NMOS transistor is on. This switch does not happen instantaneously so for a short period of time there is a direct path from the power supply to the ground and a short-circuit current appears. By careful design of the transition edges of the input signal, this current can be minimized. Furthermore, when the state switch has completed, the charge stored in the load capacitance is discharged via the NMOS transistor to the ground. This charging and discharging activity on the node capacitance is the principal component of power consumption, as power equal to $C_L V_{DD}^2$ is consumed. The factor α_1 in Equation (4.1) is called the activity factor and it represents the percentage of clock cycles where the specific gate switches states and the product $\alpha_1 f_{clock}$ is the effective switching rate of the gate.

Equation (4.1) essentially indicates the degrees of freedom available for a designer, in order to reduce power consumption of the unit under design. These are:

- *Lowering the supply voltage*:
  Voltage is the most important factor, because of its quadratic relationship to power. In addition, reducing operating voltage has a global effect, as opposed to other approaches. On the other hand, the penalty is degradation in performance. The switching delay of a CMOS gate is given as $T_d = K\frac{V_{DD}}{(V_{DD}-V_{th})^\alpha}$, where K and V_{th} are technology-related constants. It can be seen that lower voltages result in considerable bigger switching delays and therefore slower circuits. Low-voltage technologies are specially developed in order to be used to lower the power consumption.

- *Reducing capacitance of the circuit*:
  Physical capacitance in a digital circuit appears in both circuit devices and interconnection wires. By using fewer and smaller devices, as well as shorter interconnection paths, the physical capacitance is lowered. The disadvantage here is again performance degradation, as for example smaller devices have a smaller current drive capability. It is preferable, consequently, to focus on reducing the supply voltage instead.

- *Reducing nodal switching activity*
  This approach denotes either lowering the operating frequency of the circuit (having again a global effect) or lowering the activity factor of nodes that drive large capacitances. This option permits a number of design techniques to be used, which improve the power consumption of the system without compromising its performance significantly.

There are some recurring themes in the techniques employed in order to reduce power consumption. The four principal ones are *trading area/performance for power*, *adapting designs to application specific conditions* so that they just meet their functional requirements, *avoiding wasteful switching* and *exploiting locality* in order to avoid costly operations. These themes have been more or less utilized also in our design, in order to minimize power, and they will be stressed out in each particular case.

4.3 Previous work

Numerous implementation on elliptic curve cryptography exist, but few of them focus on minimizing the power consumption of the hardware. One such design is presented in [12], which is based on a non-standard protocol and field sizes in the range of [130, 140] bits in order to meet area requirements. The authors support that these key sizes provide adequate security strength according to the current state of the art. Using $0.25\mu m$ technology, their unit requires $\sim$ 14k gates and no actual power consumption figures are reported. Kumar et al. ([30]) present another design for RFID tags, with field sizes in [119,193] bits, as defined in standards from SECG and NIST. Affine coordinates are utilized for the scalar multiplication, the arithmetic unit supports operations over $GF(2^m)$ only and thus not the complete ECDSA. The unit includes dedicated modules for squaring, addition and multiplication, while inversion uses existing modules. The main target of that design was minimal area and corresponding figures range from 10k gates to 18k gates, when implemented using $0.35\mu m$ technology. Again, no power consumption figures are reported. The only design that fully supports ECDSA-compliant arithmetic operations, is the one presented in [53]. The design cannot independently generate signatures, as an external hash unit is required. However, EC operations are fully supported in both $GF(2^m)$ and $GF(p)$, through a dual-field unit and a control unit that is based on a highly-optimized 4-bit RISC microcontroller core. Projective coordinates are used to represent EC points, and Montgomery's method is used for modular multiplication, which is the only multi-cycle operation in the design. Implemented using $0.35\mu m$ technology, the design uses 23800 gates or 1.31 mm^2 for a key size of 192 bits. The estimated power consumption is $500\mu W/$MHz, which means that a clock frequency

of 100 KHz is the maximum allowed one so as to remain within the power limits. At this frequency, a scalar multiplication requires 7.1 seconds when the elliptic curve is defined over $GF(2^{191})$.

In most of the cases, the proposed design only performs part of the necessary computations for generation of the signature. For example, ECDSA compliant designs must include a hash function (and consequently a random number generator) which is usually not taken into account in the result figures. This additional functionality must be supported by an RFID tag cryptographic unit, thus making design constraints even stricter. The design of [28] of a hash unit should be reported, which implements the SHA^{-1} standards targeting low-power applications. It has the smallest area reported in the literature, roughly 4300 gates, implemented using $0.13\mu m$ technology characterized for low power. When clocked at 500 KHz, the unit consumes an average of $26.73\mu W$, and therefore it can be clocked even with the double frequency without crossing the limit. The computation takes 405 cycles or 5.72 ns at 500 KHz. These figures do not include the register file, which can have a significant area and power consumption overhead, depending on the implementation.

4.4 EllipticCore Design Choices

The higher the level of abstraction, for which a design choice is made, the higher the impact it has on the final design itself. Following this principle, a top-down design methodology was employed, selecting in each level of abstraction a choice that would help keeping power consumption levels as well as design complexity low.

In Section 4.1.1 we specify power, latency and area requirements for this design, to $\sim 50\mu W$, $[318.8, 323.3](\mu sec)$ and 1-2 mm^2 respectively. Starting from the highest possible level, the range of elliptic curves supported by EllipticCore were narrowed down to one; a single set of recommended elliptic curve parameters from [38] was selected for the implementation and specifically the ones that guarantee an implementation requiring the least resources. This choice sacrifices flexibility, but it permits extensive optimization techniques to be employed, in order to meet power and area requirements. The elliptic curve collection given in the standard is classified according to the security level it provides. The standard provides parameter values, generated by a seeded cryptographic hash function (SHA-1 in specific) in a pseudorandom fashion. Although the elliptic curve is fixed, according to the standard the base point can be arbitrarily chosen by the user, to ensure cryptographic separation of networks (and RFID tags). The selected elliptic curve is termed B-163 and the corresponding elliptic curve parameters are shown in Table 4.1. Large integers are given in hexadecimal notation.

Binary field arithmetic is more resource- and power-efficient than prime field arithmetic, hence the elliptic curve of choice is defined over a binary field. In order to avoid redundant basis transformations, polynomial basis was selected for the finite field element representation. Even though the scalar multiplication will be performed entirely using binary field arithmetic, some additional calculations need to be performed over a prime field, using the base point order r as the prime modulus. For this reason, both low-power designs (such as the ones previously reported) and low-area designs are commonly based on dual-field arithmetic units. In our design, the choice was to use separate

Parameter	Name	Value
FT	Field Type	binary extension field
p(x)	Irreducible polynomial	$p(x) = x^{163} + x^7 + x^6 + x^3 + 1$
m	size	$m = 163$
a	curve parameter	$a = 1$
b	curve parameter	$20a601907b8c953ca1481eb10512f78744a3205fd$
Gx	Base Point x	$3f0eba16286a2d57ea0991168d4994637e8343e36$
Gy	Base point y	$0d51fbc6c71a0094fa2cdd545b11c5c0c797324f1$
r	Base point order	$4000000000000000000020108a2e0cc0d99f8a5$
h	cofactor	$f = 2$

Table 4.1: Parameter values for elliptic curve B-163

computational modules for binary and prime field arithmetic. As mentioned already, the most demanding operation is the scalar multiplication. In order to simplify the logic and allow greater flexibility for optimization, a dedicated binary field arithmetic unit is chosen. A separate module will be responsible for the remaining calculations in the prime field. Requiring significantly less computational effort, these can be performed using less efficient hardware, while keeping control of power consumption figures. This approach obviously has an area penalty, which is partially ameliorated by judicious reuse of certain logic modules.

Designing the signature generation unit with a modular approach, consisting of simple modules optimized for a specific task, is advantageous also when the design of control logic is considered. ECDSA contains a large number of elementary finite field operations which lead to complex control structures. A designer might opt for a small-footprint microprocessor core, as is the case in [52]. A complex control unit is more difficult to optimize for area or power consumption and wasteful switching activity will almost certainly take place. On the other hand, a hierarchical approach offers advantages in this respect. It consists of smaller control units that control specific units modules by generating the corresponding control signals. The overall sequencing of operations is supervised by a main control unit. The main control unit is responsible for enabling the appropriate module and its control unit, according to the algorithm. The rest of the modules remain disabled, hence minimizing power consumption on the circuit. This approach permits efficient area and power optimization of each smaller control unit. In addition, each unit can be placed near the corresponding arithmetic unit's logic and therefore the wires of the control logic signals will be relatively shorter, when compared to a centralized control unit approach. This positively affects the performance figures of the circuit. In addition, placing and routing of the unit are greatly simplified.

A block diagram of the control unit of the EllipticCore module is provided in Figure 4.2 as an example. Global Signals, coming from separate modules that control shared resources, are multiplexed by the top control unit. Dedicated logic for each module is controlled by local short wires. In addition, serializing signals enable or disable modules depending on the operation required by ECDSA at a specific moment. This approach results in small separate control modules that contribute in lowering the overall power

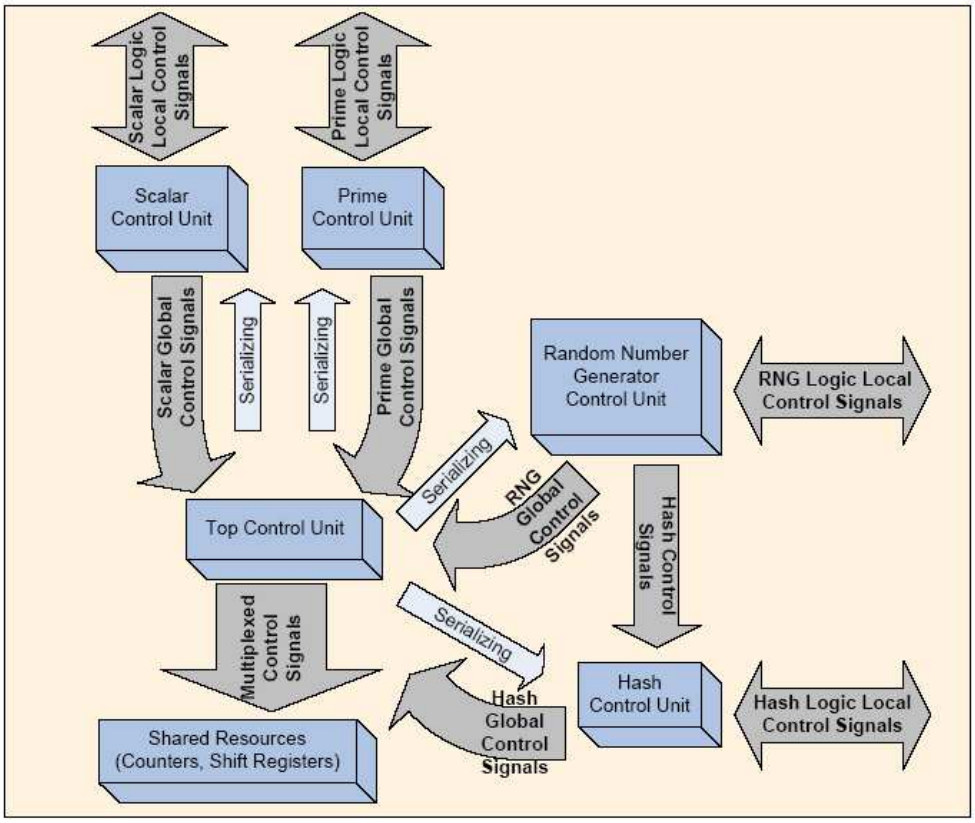

Figure 4.2: Hierarchical architecture of control unit for EllipticCore

consumption.

4.5 Implementation

The architecture of the signature-generating module can be seen in Figure 4.3. The hash function is used for generating the message digest e as well as the randomly generated nonce k. The Scalar Multiplication unit uses the nonce and the fixed base point P in order to calculate the x coordinate of $Q = k \cdot P$, which is then sent to the prime field unit for generation of the signature.

4.5.1 Hash Function

The hashing unit complies with the SHA-1 secure hash standard [3]. The algorithm computes a condensed representation of a message of maximum length 2^{64}. The output is a 160-bit long message digest e. Computations are performed on 32-bit words. The original message is segmented in 16-word blocks $(M_1, M_2, .., M_n)$ with the last one properly

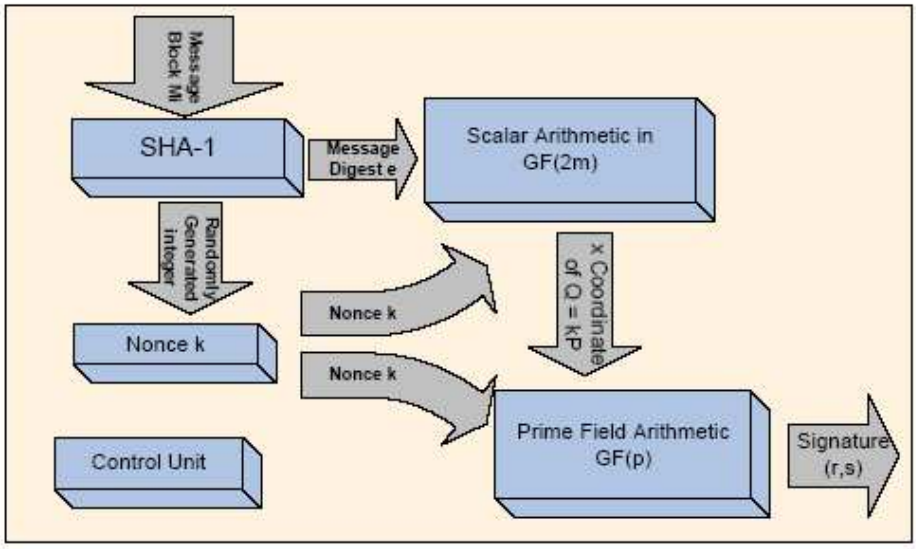

Figure 4.3: Block Diagram of the EllipticCore unit

padded with zeros, if its length does not exactly fit 16 words [1]. Message segmentation and padding are simple operations which can be easily supported by the RFID tag's control unit. The hashing unit processes the message blocks in order, and the processing of each block requires 4 *rounds* of 20 computational steps. The block's words are input sequentially to the *message scheduling unit* which computes 80 expanded words W_t to be used in the main hash computation. The latter takes place in the *message compression unit*, which eventually outputs the message digest. The expanded words are calculated according to Equation (4.2), except for $W_t, t \in [0, 16]$ which are the words of the input block. $S^n()$ denotes the rotation of the operand to the left for n positions.

$$W_t = S^1(W_{t-3} \oplus W_{t-8} \oplus W_{t-14} \oplus W_{t-16}), t \in [0, 80] \tag{4.2}$$

The message compression unit requires five 32-bit variables, which are updated in each iteration according to Algorithm (10). The constant words K_t have the following values:

- $K_t = 0 \times \texttt{5A827999}, t \in [0, 19]$
- $K_t = 0 \times \texttt{6ED9EBA1}, t \in [20, 39]$
- $K_t = 0 \times \texttt{8F1BBCDC}, t \in [40, 59]$
- $K_t = 0 \times \texttt{CA62C1D6}, t \in [60, 79]$

$f_t()$ is a logical function that operates on three 32-bit words, producing a single-word output and is defined as follows:

[1] if the message is shorter than 16 words, then it fits in a single message block, also padded with zeros to fill 16 words

- $f_t(B,C,D) = (B \cdot C) + (\bar{B} \cdot D), t \in [0,19]$
- $f_t(B,C,D) = (B \oplus C \oplus D), t \in [20,39]$
- $f_t(B,C,D) = (B \cdot C) + (B \cdot D) + (C \cdot D), t \in [40,59]$
- $f_t(B,C,D) = (B \oplus C \oplus D), t \in [60,79]$

After the 80 compression steps have been completed, the next block is processed, until and including the last block of the message. Finally, the constants K_t are added to the variables once more. The final output is formed by the content of the five variables A,B,C,D,E and corresponds to the message digest e.

Algorithm 10: Message Compression Algorithm for SHA-1

Input : $W_t, K_t, t \in [0,80]$
Output: message digest $= (A \| B \| C \| D \| E)$

$A \leftarrow 0 \times 67452301$
$B \leftarrow 0 \times EFCDAB89$
$C \leftarrow 0 \times 98BADCFE$
$D \leftarrow 0 \times 10325476$
$E \leftarrow 0 \times C3D2E1F0$
for *t = 0 to 79* **do**
    $E \leftarrow D$
    $D \leftarrow C$
    $C \leftarrow S^{30}(B)$
    $B \leftarrow A$
    $A \leftarrow S^5(A) + f_t(B,C,D) + E + W_t + K_t$
end
$A \leftarrow A + 0 \times 67452301$
$B \leftarrow B + 0 \times EFCDAB89$
$C \leftarrow C + 0 \times 98BADCFE$
$D \leftarrow D + 0 \times 10325476$
$E \leftarrow E + 0 \times C3D2E1F0$

Our implementation is largely based on the work described in [28], structurally altered in order to achieve lower power consumption. A popular implementation of the message scheduling unit is a shift register comprising sixteen 32-bit registers. The expanded words are often stored in an intermediate memory module of $80 \times 32 = 2560$ bits, before being used by the message compression unit. From a power consumption point of view, simultaneous updating of sixteen 32-bit registers is very inefficient. Furthermore, a 2.5 Kbit memory module is expensive in terms of chip area. A resource-constrained implementation opts for serialized computation of Equation (4.2) instead, where each word is fetched from a register file (of sixteen words) and added to a temporary variable. Five clock cycles are required for one expanded word to be produced. Since each expanded word is used only once in the message compression unit, we chose to directly route them to the latter instead of storing them in a costly intermediate memory module.

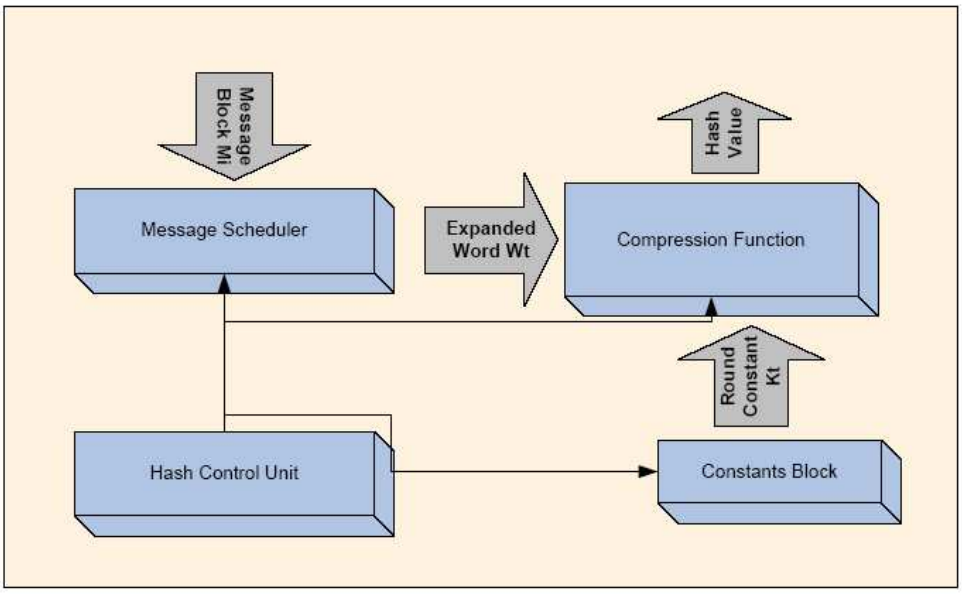

Figure 4.4: Block Diagram of the hash Unit

The message compression unit is designed to function in a synchronized fashion with the message scheduling unit, therefore requiring five clock cycles per iteration. Hardware resources are spared, as a single binary adder is used and the necessary registers are kept down to a minimum. This approach is also power-efficient, as the number of registers that are updated in each cycle is minimized to only two or three 32-bit registers. In particular, Algorithm (10) shows that each variable is used and then updated in each iteration. The computation of the next value of A requires five cycles in total, as the five terms of the corresponding formula are accumulated in each cycle one-by-one. The rest of the variables only require a single cycle to update their values. Therefore, the variable that was accumulated in a specific iteration can be updated in a subsequent cycle, as opposed to updating all variables together in one cycle. In this way, register updates are minimized to the accumulator register for A, a register holding a variable that is available for update and in some cases a register holding an intermediate value. In contrast, the design of [28] updates five 32-bit registers in a pipeline fashion, thus wasting more power.

This design may prolong the necessary time to compute one message digest, but latency is not of primary interest here. Few calls to the hashing unit are made per signature generation, and the calculation is relatively short. As an example, in most of the tag response types described in the ISO 15693 standard, tag responses are of small size, enough to fit in a single memory block. An exception is reading multiple memory blocks up to a total of 64 kbits (*Read multiple blocks* command), which spans more than one block M_i, as defined in the beginning of this section. For instance, a digital picture from an ePassport would require reading multiple blocks. For such long-latency cases, special provisions must be considered in the standard. As a conclusion, a large-latency

implementation can still be useful in an RFID tag, since low speed is a small price to pay for minimizing occupied area and power consumption.

4.5.2 Random Number Generation

Usage of the hash unit as a random number generator is described in [38]. In specific, a secret seed key is used as an input to the hash unit. The hash output is then used as the seed for another call to the hash unit. The new hash output is appended to the old one and modular reduction is then performed to bring the number to a suitable range, for the next operations where it is used (Algorithm (11)). In this case $b = 163$. Apart from the hash unit, a b-bit adder/subtractor is required in order to compute the nonce. The modulus (n-1) in the final calculation of k is also precalculated. The additional hardware resources are then a b-bit adder and two registers, for storage of the seed key and the nonce. Modular reduction of the 320-bit concatenation of x_1 and x_2 takes place at word-level, using 163-bit words and iteratively subtracting the modulus until the result is in the desired range.

Algorithm 11: Random Number Generation using SHA-1

Input : b-bit XKEY, $b \in [160, 512]$, optional user-defined seeds $XSEED_1$, $XSEED_2$
Output: b-bit nonce k

1 $XVAL \leftarrow (XKEY + XSEED_1) \bmod 2^b$
2 $x_1 \leftarrow SHA(XVAL)$
3 $XKEY \leftarrow (1 + XKEY + x_1) \bmod 2^b$
4 $XVAL \leftarrow (XKEY + XSEED_2) \bmod 2^b$
5 $x_2 \leftarrow SHA(XVAL)$
6 $XKEY \leftarrow (1 + XKEY + x_2)$
7 $k \leftarrow ((x_1 \| x_2) \bmod (n\text{-}1)) + 1$

4.5.3 Counters and Shift registers

In this design, counters are used to count the iterations in the modular division, the scalar multiplication operations (Algorithm (12) and Algorithm (7)) of the binary field module, as well as the modular multiplication and exponentiation operations (Algorithm (2) and Algorithm (3)) of the prime field module. Moreover, in these algorithms it can be seen that certain variables need to be indexed bitwise; for example in Algorithm (12) the least significant bit of register U must be processed in every iteration. These counters must count up to $M = 163$.

The choice of counter implementation in the design is of great significance, for a number of reasons. A normal binary counter, which is implemented as an adder with operands a *count value* and a constant, is a compact design with small area requirements. In addition, the output is in standard binary form, which simplifies decoding whenever the count value is used in the algorithm. On the other hand, the critical path of the circuit is long and degrades its performance. Furthermore, the transition between the

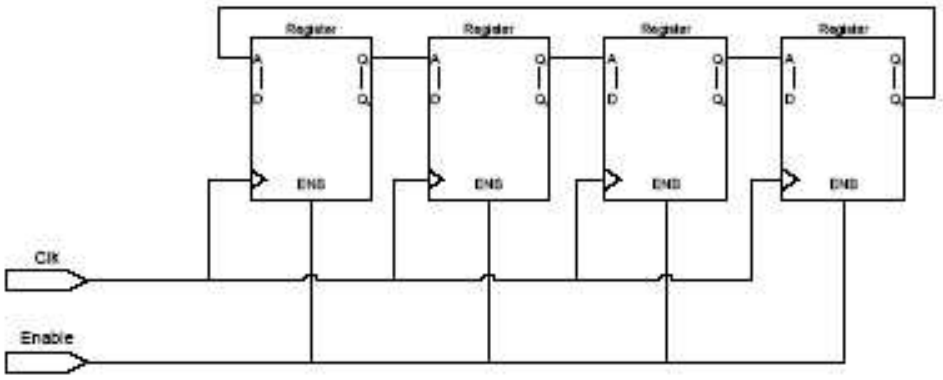

Figure 4.5: A 4-bit Johnson counter that can count 8 states

count values is not glitch-free and this fact can result in wasteful gate switching. In [50], a number of constant-time counters is presented, that is, counters with delay independent of the count value size. Naturally, the primary interest here is in low-power counter implementations.

A suitable choice of a low-power consumption counter is the *Johnson* or *twisted-tail* counter (Figure 4.5). It is a special case of a shift register, where the output from the last stage is inverted and fed back to the first stage. A Johnson Counter that counts n values, requires $n/2$ flip flops that store the counting value. The area requirements are thus much higher when compared to a normal binary counter ($\lceil log_2 n \rceil$ flip flops) and this is the penalty for using a low-power counter. Power consumption, on the other hand, is lower due to the fact that the critical path delay is equal to a single flip-flop and the state transitions are completely glitch-free. Glitches are spurious short pulses that appear due to unbalanced delay paths in the computational logic that updates the count value of a counter. Such glitches might affect the operation of the unit, leading to erroneous results and wasteful energy consumption.

Another benefit of using Johnson counters, is that a few gates are required to generate an address signal, useful for the variable indexing mentioned above. Johnson counters are inherently able to count even numbers. Since odd count numbers are also required, a modified Johnson counter has been developed, with an additional flip-flop for the extra count state. When the counter reaches this extra state, it is reset to the initial value. For the bitwise indexing of variables in the design, a shift register would be normally used. However, an approach more suitable for low-power designs keeps the contents of the register fixed and uses a mask to select the desired bit. This addressing signal is generated by the Johnson Counter design and it contains M-1 zero bits and a non-zero one, indicating the bit to be indexed next. A bitwise OR operation of the masked contents of the register produces the output bit. As a result, instead of m flip-flops switching (the worst-case of a shift register), one two-input AND gate and $log_2 k$ 2-input OR gates switch at each moment. These benefits largely justify the usage of Johnson

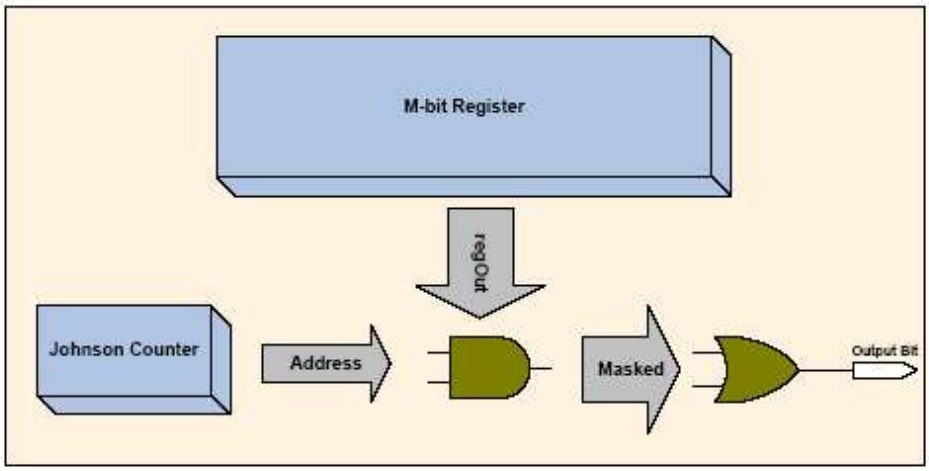

Figure 4.6: Implementation of a low-power shift register

counters for state counting and address signal generation in order to achieve lower power consumption levels.

4.5.4 Scalar Multiplication Module

For the scalar multiplication operation, Montgomery's Ladder is a very attractive choice, as described in Chapter 3. In specific, a modified version of Algorithm (7) is employed in our design. The unit comprises an arithmetic unit and a control unit and it also makes use of counters that are shared with the prime field arithmetic unit, as in both cases the same number of count states are required.

It was observed that the ECDSA algorithm does not require the complete scalar multiplication to be performed. In specific, only the x-coordinate of the resulting point is needed, for the computations in lines 3-8 of Algorithm (8). Consequently, the last part of the scalar multiplication algorithm, retrieving the y-coordinate in lines 12-13, are redundant and can be omitted resulting in a simpler arithmetic and control unit. Usage of affine coordinates has the advantage that a smaller number of finite field operations is computed, as formulas used for point addition and point doubling are less complicated. As a result, the control unit that helps serializing the required finite field operations is more compact.

As can be seen in Algorithm (7), two modular divisions are performed per iteration. A sensible choice is then to base the design of the scalar multiplication unit on a modular divider in GF(2^m), with additional logic in order to capacitate it to perform additions and squaring operations in the same field. The radix-2 algorithm proposed in [17] takes 2m-1 cycles to calculate one modular division. The radix-4 version requires additional digital logic and a more complicated control unit to calculate it in half the cycles. Since the primary optimization goal is not performance, the simple bit-serial modular divider is used. This algorithm is based on the binary extended Euclidean algorithm, incorporating the improvements mentioned in Section 3.1.2.2. The exact modified version, as used in

Algorithm 12: Bit-serial modular division over GF(2^m)

Input : X,Y $\in GF(2^m)$, p
Output: (X/Y) mod p

$U \leftarrow Y$
$V \leftarrow p$
$R \leftarrow X$
$S \leftarrow 0$
k=2m-2
$D \leftarrow 1$
$IsPos \leftarrow 1$
while $k \geq 0$ **do**
    if $u_0 = 0$ **then**
        $U \leftarrow U/2, R \leftarrow R/2$ mod p
        if *IsPos = 0* **then**
            $D \leftarrow D + 1$
        else if *D = 0* **then**
            $D \leftarrow D + 1, IsPos \leftarrow 0$
        else
            $D \leftarrow D - 1$
        end
    else if *IsPos = 1* **then**
        $U \leftarrow U/2 \oplus V/2, R \leftarrow R/2 \oplus V/2$ mod p
        if *D = 0* **then**
            $D \leftarrow D + 1, IsPos \leftarrow 0$
        else
            $D \leftarrow D - 1$
        end
    else
        $D \leftarrow D - 1, IsPos \leftarrow 1$
        $U \leftarrow U/2 \oplus V/2, V \leftarrow U$
        $R \leftarrow R/2 \oplus S/2$ mod p, $S \leftarrow R$
    end
end
return S

the present implementation, is found in Algorithm (12). Four state registers (U,V,R,S) are used, with only two of them being updated in every cycle. In order to lower the instantaneous power consumption, the state registers are grouped in pairs (U,V) and (R,S) that are clocked by two clocks of opposite phase. This way, two registers at max update their values per clock edge within the unit. Addition and squaring are performed according to the corresponding algorithms presented in Section 3.1.2.2. The design is customized for use of the particular binary field and therefore imposes a minimal overhead when compared to similar arithmetic units that support arbitrary binary fields.

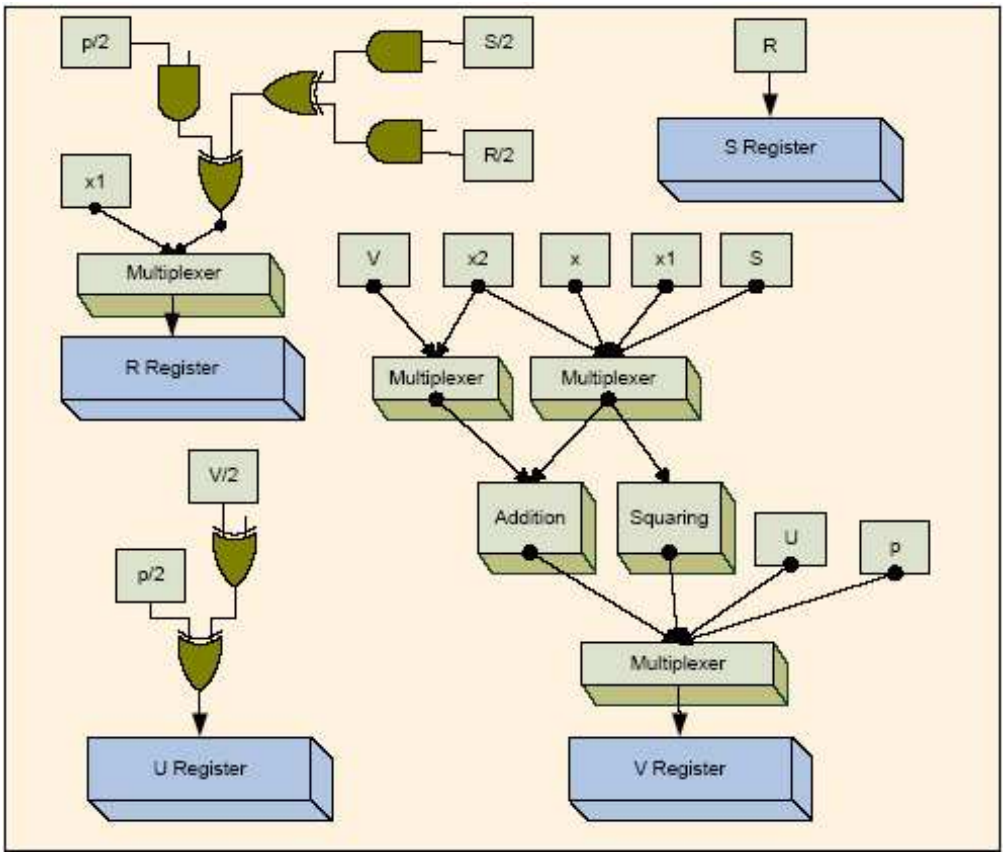

Figure 4.7: Block Diagram of the Scalar Arithmetic Unit

For the complete scalar multiplication algorithm, two more registers are used to store intermediate values, as well as the final results. The design is shown in Figure 4.7. One detail that was left out of the diagram for clarity, is some extra gating logic that selectively enables the signals that are fed into the combinational blocks. Gating is necessary in order to keep the inputs fixed, for combinational blocks that are not in use during a specific cycle. For example, the addition and squaring blocks are not used throughout modular division. This way, wasteful switching within the blocks is avoided. This low-power design technique is known as *operand isolation* and the main idea is further explained in [37]. Instead of gates, latches have been used in other designs, in which case more granular signal gating is possible, at the expense of additional area, as explained in [13]. For this design though, gating through AND gates is adequately efficient. The gating signals are controlled by the Control Unit.

An up/down signed counter (of size $\lfloor log_2(k) \rfloor + 1$) is required, which is implemented as an incrementer/decrementer, a customized adder with one operand being 1/-1 (*dCounter*), along with a flip-flop indicating the sign of the counting value (*IsPos*). It should be noted that the exact count value is of no interest for the division algorithm. Only the sign and a zero value are considered. Another counter is necessary for counting the $2m$-2 iterations of the algorithm. This is implemented as a Johnson counter, as de-

scribed in Section 4.5.3. As can be seen in Algorithm (4), each bit of the scalar k needs to be processed separately. For this purpose, the shift register shown in Figure 4.6 is used.

4.5.5 Prime field Module

This unit is responsible for performing the calculations in lines 3-8 of Algorithm (8). In other words, it must support modular reduction, modular inversion, modular multiplication and modular addition in $GF(n)$, with n being the base point order mentioned in Table 4.1. The relevant algorithms are the ones presented in Section 3.1.2.1. Since these operations occur once per signature and are relatively less time consuming, this unit can employ less efficient design choices, in order to reduce occupied area and power consumption. The most prominent example is modular reduction, which is performed in a naive way of subtracting the modulus until the result is of smaller magnitude. If the result is of positive sign, another subtraction takes place. A final correction step is necessary, because one subtraction too many takes place. A maximum of three subtractions are necessary; since operands are in the range $[0, n)$, an addition will result in a number in the range $[0, 2n)$ which requires at maximum only one subtraction of the modulus to be reduced. According to the modular multiplication algorithm (Algorithm (2)) for the prime case, one left shift operation and one addition are performed, and the result is in the range $[0, 3n)$. Including the last test subtraction (which will be compensated by the addition of the correction step) leads to a total of three subtractions and four cycles in the worst case. Performance-oriented designs would rather choose a single-step reduction circuit instead, which would however be more costly power-wise.

A single binary adder is used for all the additions, in order to save area and power. Although the critical path is severely hampered, a ripple-carry adder introduces the least area overhead and is the most power-efficient choice among binary adders, as is experimentally concluded in [44]. Negative numbers are represented in their 2's complement form, therefore the sign test for the modular reduction operation is performed by checking the most significant bit of the result, as indicated by Equation (3.1). Since the modulus that is added/subtracted is fixed, both the positive and the negative moduli are hardwired to save area. For the modular multiplication, apart from the addition, a single-step left shift operation is required, which is implemented in a shift register fashion.

Modular inversion is not performed via the costly binary extended Euclidean algorithm equivalent for prime fields, but rather with the one based on Fermat's little theorem, which is implemented using little more than already existing resources. The Johnson counters mentioned in Section 4.5.3 are used here as well, the former for accessing the exponent's bits in the modular exponentiation algorithm of 3 and the latter for accessing the multiplier's bits during all the modular multiplications. Finally, three registers are used in total, one for storing the operation result, one for storing the operand and an additional scratch register for storing intermediate results. The first one is appropriately sized to fit the intermediate results of the modular multiplication operation, without having overflow phenomena. The layout of the unit is depicted in Figure 4.8.

The prime field arithmetic unit takes up significantly less resources when compared

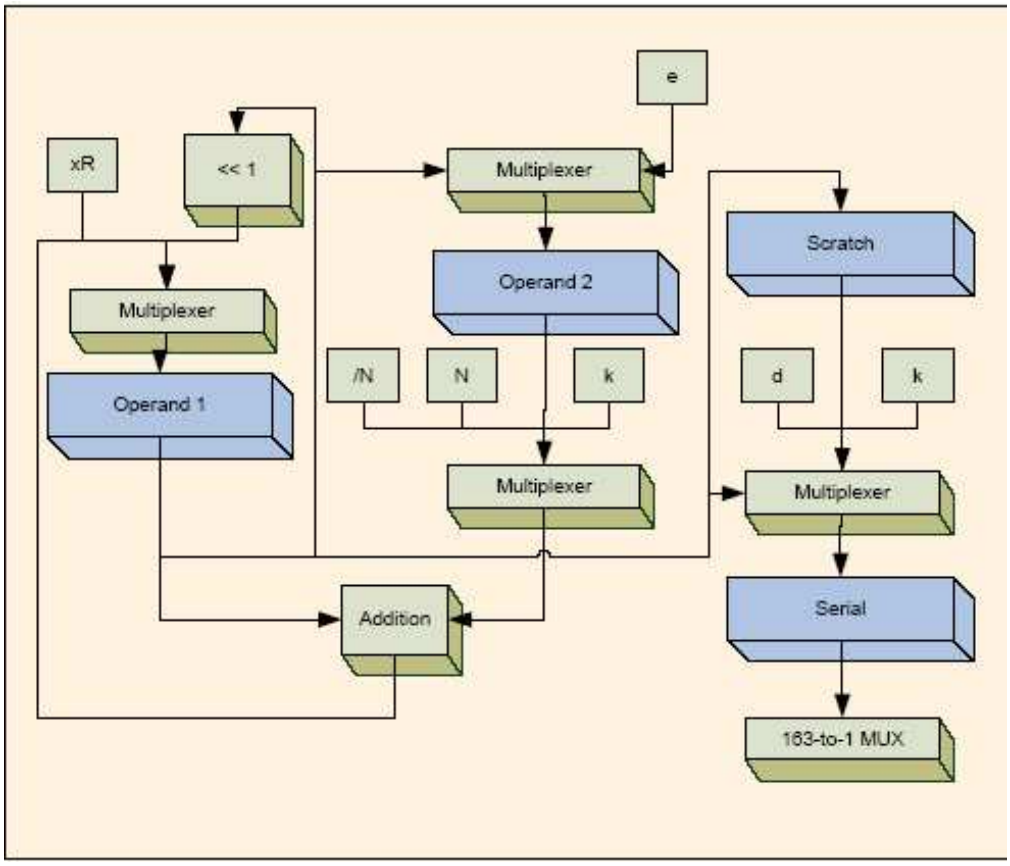

Figure 4.8: Block Diagram of the Prime Field Module

to the scalar multiplication unit. The implemented algorithms are not as efficient, therefore it takes time almost equal to that of the scalar arithmetic unit to complete the calculations, even though the number of elementary field operations involved is significantly smaller. In order to further reduce power consumption of the unit, it is possible to clock the unit with half the system frequency (using a simple frequency divider) without significantly affecting the overall EllipticCore performance.

4.6 Design flow and Evaluation methodology

In the following paragraphs, the methodology that was followed to verify and evaluate the design is described, as well as the tools used during the design process.

4.6.1 Evaluation Methodology

After selecting the best-fitted algorithms for the design, a reference software implementation was developed, based on C libraries for arithmetic in $GF(2^{163})$ (URL: http://www.beautylabs.net/software/finitefields.html) and on the GNU Multi-precision library for GCC [5], for arithmetic in $GF(p)$. These libraries greatly simplified the devel-

opment of the chosen algorithms. Although modular arithmetic operations are directly supported by the libraries, the algorithms were separately implemented in order to verify the hardware model in a step-by-step fashion.

The corresponding hardware model was developed using the VHDL language ([23]) at the Register Transfer Level (RTL) of abstraction. In RTL design, a circuit's behavior is defined in terms of the flow of signals or transfer of data between registers, and the logical operations performed on those signals. The software verification model was used to compare results with the hardware model, both at the end of the computation as well as in algorithm iterations separately. Especially for the hash unit, the standard itself contains test runs, which were used for verification of the models, and thus no software model needed to be developed.

Subsequently, a structural model of the hardware version was developed and again verified against the software model. The structural model was in a form that a synthesis tool can understand and map to a specific technology. It includes implementation specifics (such as the forms of counters and the arithmetic circuits) that are supplied so that the output of the synthesis tool is of acceptable performance. Finally, power consumption measurements were acquired using the suitable synthesizer tools, as well as figures for the other performance metrics (area and clock count) and compared with the original design constraints set in paragraph 4.1.1.

4.6.2 Design Flow

The different modules of EllipticCore were initially designed to support the full arithmetic functionality and then they were stripped down for optimization purposes. For instance, the scalar multiplication unit was first implemented to perform the full scalar multiplication. As is pointed out in Section 4.5.4, the y-coordinate of the resulting EC point is not necessary, therefore the redundant functionality was removed eventually, including modular multiplication in the binary field. The chosen tool for code development was Emacs, enhanced with support for VHDL and the simulations were performed with *Modelsim 6.1f* from Modeltech, one of the standard functional simulation tools in the industry.

The structural hardware model was compiled with *Design Compiler* from Synopsys Corporation, another industry-standard tool. Design Compiler translates the structural model to a gate-level description of the system. As a first step, the tool selects gates from a generic library (GTECH) and subsequently, using gates from a user-specified technology library, the final mapping is performed. The technology library used for this purpose is the $0.35\mu m$ CMOS library from Austria Micro Systems. This library was selected as the most efficient in terms of power-consumption and area, among the available libraries. It is not a library characterized for low-power, but a standard technology library. Usage of a more suitable technology library will almost certainly yield better results.

Design Compiler generates a gate-level netlist in VHDL format, along with a file in *Standard Delay Format* (SDF) which contains information on the timing delay of each component in the circuit. These two files can be used for accurate calculation of the power consumption of the circuit. Although the tool itself provides power consumption reports, by specifying the operating clock frequency, the latter are only based on statis-

Table 4.2: Power Consumption Performance with $0.35\mu m$, 3.3V Technology

Unit	**Dynamic Power** (μW)	**Leakage Power** (μW)
Scalar and Prime Field Units	78.3964	1.1342
Hash Unit	43.0904	0.471

tical estimations of the switching activity in each node of the circuit. A more accurate power figure can be obtained by capturing the actual switching activity during a precision simulation. The aforementioned netlist and the sdf file are used for this purpose, in combination with Modelsim. The captured switching activity is saved in a *Switching Activity Interchange Format* (SAIF) file. The latter is used by the Synopsys suite of tools, to annotate the switching activity to the nets of the netlist. Exact power readings are then obtained through *Power Compiler* from the same company, using Equation (4.1). It also provides figures on the power consumption due to leakage currents. Using the supplied information, the tool attempts to further optimize the design in terms of power consumption, using standard low-power techniques such as clock gating, operand isolation and efficient Finite State Machine encoding. Unfortunately, only average power consumption figures are supplied.

4.7 Results and comparison to related work

The modules described in the previous paragraphs, are the most important elements of a co-processor that can compute an ECDSA-compliant signature. In the following tables, the performance of these units is provided, in terms of speed, area and power consumption. As it has been pointed out previously, the primary goal is to minimize the power consumption of the design, in order to meet the 50 μW limit set in 4.1.1. Furthermore, its performance in terms of area and latency must still remain within acceptable limits. The following tables exhibit the design's actual measured performance as indicated by Power Compiler, with the clock frequency set at 100 KHz and a supply voltage of $V_{DD} = 3.3V$. It can be seen that the performance is unacceptable, with the specific operating conditions. On the other hand, by lowering the supply voltage, acceptable power consumption figures can be obtained. The quadratic relationship of dynamic power to voltage, according to Equation (4.1) means that halving the supply voltage yields the quarter of the power consumption figure mentioned in the table. Or, equivalently, a higher operating frequency can be used without exceeding the power threshold. In fact, a supply voltage of $V_{DD} = 1.8V$ allows doubling the operating frequency and $V_{DD} = 1.2V$ allows quadrupling the frequency. This is important when signature calculations need to be performed faster. Lowering the supply voltage is standard practice for reducing power consumption in a design and, although it deteriorates performance, this is not a crucial concern here. Further improvement on the power consumption can be obtained by utilizing a more suitable target library, with smaller feature sizes and components characterized for low-power.

As far as the occupied chip area is concerned, the design requirements set a limit of 1-2 mm^2. However, table 4.5 indicates that the current implementation exceeds the

Table 4.3: Power Consumption Estimation with $0.35\mu m$, 1.8V Technology

Unit	**Dynamic Power** (μW)	**Leakage Power** (μW)
Scalar and Prime Field Units	23.325	0.338
Hash Unit	12,821	0.14

Table 4.4: Power Consumption Estimation with $0.35\mu m$, 1.2V Technology

Unit	**Dynamic Power** (μW)	**Leakage Power** (μW)
Scalar and Prime Field Units	10.366	0.1499
Hash Unit	5,698	0.0624

desired limit. The existence of a hash unit makes things worse. The area figure of this hash unit implementation is significantly larger than the one in [28]. Contrary to the latter though, the design presented here includes the area of a large register file. On the whole, area was a performance indicator of secondary importance for this design and we were eager to sacrifice additional area in order to lower the power figure. The area figure can be improved by carefully redesigning the unit in order to reuse some resources in a more efficient way. However, the greatest improvement would be obtained by using a more advanced technology process, with smaller feature sizes. The technology we used in this work has a transistor channel length of 0.35 μm, while technologies with smaller feature sizes (such as 0.18 and 0.12 μm processes) are also available in the market.

A reliable source for estimations on the effect of future technology trends on occupied area is ITRS (International Technology Roadmap for Semiconductors). This document is updated every year to reflect the semiconductor industry's needs in research and development in a 15-year horizon. It presents industry-wide consensus on the trends in various aspects of electronics. According to [25], the observed trend for transistors in 2007 is the shrinking of the channel length by 29% every 3 years. This trend refers to technologies with the smallest possible transistor footprint available at that moment. Some products however, including RFID tags, opt for cheaper technologies with larger transistors in order to keep implementation costs low. In other words, these products use technologies with the smallest transistor size available 2-3 years before the time of implementation. According to ITRS, the same trend holds also for this category of products.

As a result, the projected occupied area for this design if a low-cost technology of 2010 was chosen for the implementation would be smaller by a factor of 50.4%. The results shown in table 4.5 indicate that the occupied area would be acceptable in that case. By moving to high-end technologies, the obtained results would be even better. However, the manufacturing costs in that case could become forbiddingly high, making thus RFID less viable.

Table 4.6 gives results on the time taken to complete the computations for each of the units, as obtained from the cycle-accurate structural model via simulation. The entry *Total* corresponds to the time required for a computation of the signature, provided

Table 4.5: Current and Projected Area Performance

Unit	**Area** (μm^2)	**Projected Area** (μm^2)
Scalar and Prime Field Units	1589144.5	800928.8
Hash Unit	610022.5	307451.3

Table 4.6: Cycle count of the different units

Unit	**Cycles**	**Time** (f_{clk} = 100 kHz)
Hash Unit	408	4080 μs
Scalar Unit	105448	1054480 μs
Prime Field Unit	120162	1201620 μs
Total	225610	2.256 s

that the memory digest and the nonce are already computed. Since three calls to this function are made in total (one for the computation of the message digest and two for the computation of the nonce), calculating the memory digest and the nonce would cost an extra 1224 μs. It can be seen that the hash operation is significantly faster than the rest of the computations. A signature computation requires 2.256 seconds, with a clock frequency of 100 KHz. This frequency guarantees that power consumption will remain below the required level.

This design is significantly faster than other low-power designs mentioned above (e.g. the design presented in [53] requires roughly double the amount of clock cycles, for the scalar multiplication alone). This is due to the simpler formulas used to perform the scalar multiplication (using affine coordinates) and the existence of the dedicated modular division module. However, the timing limits given in Section 2.1.2 for the ISO 15693 standard are obviously violated, especially when such low clock frequencies are used. Unfortunately there is little room for improvement under the given power consumption constraint.

Moving to more advanced technologies (technology processes with a minimum feature size of 180 nm and 90 nm are available in the industry) will allow an increase of the clock frequency and therefore a reduction the computation time. Another possible improvement would be pre-calculating some input values, such as the message digest for a number of frequently used tag responses. This approach has already been mentioned in Section 3.2. Finally, [53] suggests adapting the protocol layer in order to facilitate such time-consuming computations for security enhancement, by interleaving requests and responses.

Conclusions and Future Outlook

5

5.1 Conclusions

This work investigated different possibilities for enhancing the level of security in RFID applications. Depending on the security objective that is important in a particular RFID application, hardware components were designed and implemented to be used within the framework of real-life RFID protocols. In order to fit the components in a real-world application scenario, the widely used RFID protocol (ISO 15693) was selected. The detailed operation of the protocol was presented, as well as the timing restrictions posed on the communication between tags and readers.

Subsequently a device was presented, that can provide a privacy-protection mechanism by exploiting the specific protocol's operation for this purpose. The device is intended to be used as a generic platform for more complex devices, such as PDAs, which are responsible for determining when the privacy protection mechanisms are to be activated. If necessary, a jamming signal is sent to any unauthorized RFID readers that attempt to access RFID tags without their owner's permission. This device was implemented as a prototype and verified for proper operation.

Furthermore, the elliptic curve digital signature algorithm was presented in detail and considered for possible integration in RFID tags, where digital signature schemes are applicable. Subsequently, a component that can generate signatures compliant with ECDSA was presented. This component was designed with the stringent power consumption restrictions of an RFID tag in mind, sacrificing speed of calculation (primarily) and chip area (secondarily) in order to meet the restrictions. Our component provides very limited support to the ECDSA standard, in the sense that not the complete set of functionality is supported, but enough to compute valid signatures. The security level it provides is adequate for short-term security requirements. As expected, stripping the component from all generic (and thus redundant) functionality, improves power consumption figures significantly. The most important design choices include:

- using affine coordinates for the elliptic curve point representation, in order to simplify the modular arithmetic operations sequence,

- using separate resources for the scalar multiplication and the rest of the modular operations, in order to optimize the scalar multiplication,

- using a modular divider for the most demanding operation in the scalar multiplication to improve efficiency,

- using a modular finite state machine approach to obtain a simple and low-power control unit and

- customizing as many operations as possible for computations on a specific elliptic curve. The operations that benefit more from this customization are modular reduction and modular squaring, which become single-cycle operations as opposed to the multicycle ones of the generic counterpart.

EllipticCore, implemented in a standard $0.35\mu m$ CMOS process, meets the power consumption requirements while outperforming other designs with comparable power consumption levels. Furthermore, the occupied area remains relatively close to the smallest designs in the literature. Interestingly, there is still significant margin for improvement, especially if we move to a more advanced CMOS process. In conclusion, elliptic curve cryptography is applicable in the context of RFID applications. While tags with signature generating capabilities will be costly to implement, this will certainly change when the cost of using advanced processed becomes advantageous for commercial applications. It is though necessary for corresponding RFID protocols such as ISO 15693 to adapt their timing restrictions to permit these computationally intensive security operations.

Apart from the actual design effort, a significant percentage of the time spent on the thesis was devoted in selecting and setting up the equipment required for design purposes. Prototype development required becoming familiar with equipment such as oscilloscopes, impedance analyzers, signal generators and spectrum analyzers, as well as technical skills such as soldering, antenna design and implementation, prototype board design etc. Analog design tools such as Protel and Orcad were used for both schematic design and analog circuit simulation, and tools for device driver development such as the ImageCraft ICC integrated development environment for AVR microcontrollers and the Linux GCC and GDB tools for the PXA270 microprocessor. For the elliptic curve cryptography module, Modelsim was used for RTL/gate-level simulation and the Synopsys synthesis tools were used for synthesis and optimization. The compilation of the code, as well as simulation and synthesis of the models were script-based procedures, because manually entering the necessary commands for each tool would result in an exhausting and time-wasting routine.

5.2 Future Outlook

Even though the designs presented in this work meet the initial requirements, future contributions will certainly improve their performance or even enable incorporating additional functionality. The platform for the RFID Guardian should be implemented in a single board, connected via a standard interface (such as PCMCIA) to the microprocessor running the protocol algorithms (e.g. a PDA). In addition, the possibility of unifying the analog front-end for the reader and the tag part, to save space and components should be investigated. Finally, an output power control circuit, that will adjust the

transmitted power depending on the strength of the field at the location of the device would be most useful for prolonging its battery life.

Useful contributions to the second component presented here would include efforts to reduce the chip area of the design, always with care not to exceed the power consumption levels. This could be achieved through smarter reuse of available resources, such as registers. The latter are now interspersed around the module and could be collected in a register file instead. In addition, a designer with greater familiarity with the Synopsys tools would certainly optimize the design more efficiently. Finally, using a technology library more suitable for low-power designs has already been pointed out. The resulting implementation will improve the area and power consumption performance figures, which are the most important ones in this case.

The ultimate target would be autonomous tags that adequately fulfill all the security objectives mentioned in Section 1.2. However, this target remains dependent on technology and cost restrictions but compact and strong security mechanisms such as elliptic curve cryptography, combined with clever design approaches will make this target feasible in the near future.

Bibliography

[1] *eCos Home Page*, http://ecos.sourceware.org/.

[2] *On a community framework for electronic signatures*, Official Journal of the European Communities **L13** (1999), 12–20, Directive 1999/93/EC.

[3] *Secure hash standard*, National Institute of Standards and Technology, Washington, 2002, http://csrc.nist.gov/publications/fips/. Note: Federal Information Processing Standard 180-2.

[4] *A study of the energy consumption characteristics of cryptographic algorithms and security protocols*, IEEE Transactions on Mobile Computing **5** (2006), no. 2, 128–143, Student Member-Nachiketh R. Potlapally and Member-Srivaths Ravi and Senior Member-Anand Raghunathan and Fellow-Niraj K. Jha.

[5] *The GNU Multiple Precision Arithmetic Library*, 4.2.1 ed., May 2006, http://www.swox.com/gmp/gmp-man-4.2.1.pdf.

[6] ISO/IEC FDIS 15693, *Identification cards - Contactless integrated circuit(s) cards, Vicinity cards*, 2000.

[7] ASC-X9 Accredited Standards Committee X9 Inc., *Public key cryptography for the financial services industry: The elliptic curve digital signature algorithm (ECDSA)*, ANSI X9.62-1998, July 1999.

[8] Katherine Albrecht and Liz McIntyre, *RFID Nineteen Eighty Four*, http://www.spychips.com.

[9] Ross J. Anderson, *Security engineering: A guide to building dependable distributed systems*, John Wiley & Sons, Inc., New York, NY, USA, 2001.

[10] ATMEL, *8-bit AVR Microcontroller with 32 kBytes In-System Programmable Flash Data Sheet*, April 2006, http://www.atmel.org/dyn/resources/prod_documents/doc2503.pdf.

[11] Mohan Atreya, Benjamin Hammond, Stephen Paine, Paul Starrett, and Stephen Wu, *Digital signatures*, first ed., RSA Press, 2002.

[12] L. Batina, J. Guajardo, T. Kerins, N. Mentens, P. Tuyls, and I. Verbauwhede, *An elliptic curve processor suitable for rfid-tags*, Cryptology ePrint Archive, Report 2006/227, 2006, http://eprint.iacr.org/.

[13] Frank Bouwens, *Power and performance optimization for adres*, Master's thesis, Delft University of Technology, 2006.

[14] Richard P. Brent and H. T. Kung, *Systolic vlsi arrays for polynomial gcd computation.*, IEEE Trans. Computers **33** (1984), no. 8, 731–736.

[15] C.K.Koc and C.Y.Hung, *A fast algorithm for modular reduction*, IEE proceedings. Computers and digital techniques, vol. 145, Institution of Electrical Engineers, Stevenage, ROYAUME-UNI, July 1998, pp. 265–271.

[16] Certicom Corp, *Certicom announces elliptic curve cryptosystem challenge winner*, Press release, 2004, http://www.certicom.com/index.php?action=company,press_archive&view=121.

[17] Guerric Meurice de Dormale and Jean-Jacques Quisquater, *Novel iterative digit-serial modular division over GF(2m)*, CRyptographic Advances in Secure Hardware – CRASH 2005, 2005.

[18] Stephen E. Eldridge and Colin D. Walter, *Hardware implementation of montgomery's modular multiplication algorithm*, IEEE Trans. Comput. **42** (1993), no. 6, 693–699.

[19] Klaus Finkenzeller, *Rfid handbook: Fundamentals and applications in contactless smart cards and identification*, John Wiley & Sons, Inc., New York, NY, USA, 2003.

[20] ICAO New Technologies Working Group, *Biometrics deployment of machine readable travel documents*, Technical Report version 2.0, International Civil Aviation Organization (ICAO), 2004.

[21] Darrel Hankerson, Alfred J. Menezes, and Scott Vanstone, *Guide to elliptic curve cryptography*, Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2003.

[22] IDTechEx, *Rfid market* 2.77*billionin*2006*to12.35 billion in 2010*, Web Article, 2006, http://www.idtechex.com/products/en/articles/00000409.asp.

[23] IEEE, *Ieee standard vhdl language reference manual*, 2002 ed., IEEE Std 1076-2002.

[24] Intel, *Intel PXA27x Processor Family Developer's manual*, January 2006, ftp://download.intel.com/design/pca/applicationsprocessors/manuals/280000003.pdf.

[25] International Roadmap Committee, *The international technology roadmap for semiconductors*, Web Article, 2007, http://www.itrs.net/Links/2007ITRS/Home2007.htm.

[26] Tetsuya Izu and Tsuyoshi Takagi, *A fast parallel elliptic curve multiplication resistant against side channel attacks*, PKC '02: Proceedings of the 5th International Workshop on Practice and Theory in Public Key Cryptosystems (London, UK), Springer-Verlag, 2002, pp. 280–296.

[27] Anantha Chandrakasan Jan M. Rabaey and Borivoje Nikolic, *Digital integrated circuits, 2nd edition*, 2nd ed., Prentice Hall, December 2002.

[28] Jens-Peter Kaps and Berk Sunar, *Energy comparison of AES and SHA-1 for ubiquitous computing*, Embedded and Ubiquitous Computing (EUC-06) Workshop Proceedings (Xiaobo Zhou et al., ed.), Lecture Notes in Computer Science (LNCS), Springer, 2006, to appear.

[29] Neal Koblitz, *Elliptic curve cryptosystems*, Mathematics of computation **48** (1987), no. 177, 203–209.

[30] Sandeep Kumar and Christof Paar, *Are standards compliant elliptic curve cryptosystems feasible on RFID?*, Printed handout of Workshop on RFID Security – RFIDSec 06, July 2006.

[31] Julio Lopez and Ricardo Dahab, *Fast multiplication on elliptic curves over GF(2 m) without precomputation*, Cryptographic Hardware and Embedded Systems, no. Generators, 1999, pp. 316–327.

[32] E. D. Mastrovito, *VLSI architectures for computations in Galois fields*, Ph.D. thesis, Linköping University, Linköping, Sweden, 1991, p. 249.

[33] Melexis, *MLX90121 13.56 MHz RFID Transceiver Data Sheet*, 6.0 ed., December 2005.

[34] Melexis, *Transceiver RFID 13.56 MHz MLX90121 Cookbook*, 1.0 ed., June 2006.

[35] Victor S Miller, *Use of elliptic curves in cryptography*, Lecture notes in computer sciences; 218 on Advances in cryptology—CRYPTO 85 (New York, NY, USA), Springer-Verlag New York, Inc., 1986, pp. 417–426.

[36] Peter L. Montgomery, *Modular multiplication without trial division*, Mathematics of Computation **44** (1985), 519–521.

[37] M. Munch, N. Wehn, B. Wurth, R. Mehra, and J. Sproch, *Automating rt-level operand isolation to minimize power consumption in datapaths*, date **00** (2000), 624.

[38] National Institute of Standards and Technology, *FIPS PUB 186-2: Digital Signature Standard (DSS)*, January 2000.

[39] NXP Semiconductors, *High Sensitivity Applications of Low-Power RF/IF integrated circuits*, August 1997, http://www.standardics.nxp.com/support/documents/rf/pdf/an1993.pdf.

[40] Institute of Electrical and IEEE Electronics Engineers, *Standard specifications for public key cryptography*, IEEE 1363-2000, 2000.

[41] Rolf Oppliger, *Internet and intranet security, second edition*, Artech House, Inc., Norwood, MA, USA, 2001.

[42] Behrooz Parhami, *Computer arithmetic: Algorithms and hardware designs*, 2000.

[43] PolyIC, *Polyic announces printable rfid prototypes*, Web Article, 2006, http://www.rfidupdate.com/articles/index.php?id=1213.

[44] Jan M. Rabaey and Massoud Pedram, *Low power design methodologies*, The International Series in Engineering and Computer Science, no. 336, Kluwer Academic Publishers, 1995.

[45] Thomas Ricker, *Dutch RFID e-passport cracked – US next?*, http://www.engadget.com/2006/02/03/dutch-rfid-e-passport-cracked-us-next/.

[46] Melanie R. Rieback, Bruno Crispo, and Andrew S. Tanenbaum, *RFID guardian: A battery-powered mobile device for RFID privacy management*, Proc. 10th Australasian Conf. on Information Security and Privacy (ACISP 2005), LNCS, vol. 3574, Springer-Verlag, July 2005, pp. 184–194.

[47] R. L. Rivest, A. Shamir, and L. Adleman, *A method for obtaining digital signatures and public-key cryptosystems*, Commun. ACM **26** (1983), no. 1, 96–99.

[48] Philips Semiconductors, *SA605 : High Performance low power mixer FM IF system*, November 1997, Data Sheet.

[49] Leilei Song and Keshab K. Parhi, *Low-energy digit-serial/parallel finite field multipliers*, J. VLSI Signal Process. Syst. **19** (1998), no. 2, 149–166.

[50] Mircea R. Stan, Alexandre F. Tenca, and Milos D. Ercegovac, *Long and fast up/-down counters*, IEEE Trans. Comput. **47** (1998), no. 7, 722–735.

[51] Xiaoyun Wang, Yiqun Lisa Yin, and Hongbo Yu, *Finding collisions in the full SHA-1*, **3621** (2005), 17–28.

[52] Johannes Wolkerstorfer, *Hardware aspects of elliptic curve cryptography*, Ph.D. thesis, Graz University of Technology, 2004.

[53] Johannes Wolkerstorfer, *Is elliptic-curve cryptography suitable to secure RFID tags?*, Handout of the Ecrypt Workshop on RFID and Lightweight Crypto, July 2005.

Curriculum Vitae

Dimitris Stafylarakis has completed the five-year curriculum of the Electrical and Computer Engineering Department at the University of Patras in Greece. Subsequently, he followed the two-year Master of Science program at the Computer Engineering group of Delft University of Technology. He is currently working as a software engineer for a dutch consultancy company.