CRYPTOGRAPHIC KEY DISTRIBUTION IN WIRELESS SENSOR NETWORKS: A HARDWARE PERSPECTIVE

by

Kealan McCusker, B.Sc. M.Sc.

Submitted in partial fulfilment of the requirements

for the Degree of Doctor of Philosophy

Dublin City University

School of Electronic Engineering

Supervisor: Dr. Noel E. O'Connor

January, 2008

I hereby certify that this material, which I now submit for assessment on the programme of study leading to the award of Doctor of Philosophy is entirely my own work, that I have exercised reasonable care to ensure that the work is original, and does not to the best of my knowledge breach any law of copyright, and has not been taken from the work of others save and to the extent that such work has been cited and acknowledged within the text of my work.

Signed: _____
Kealan McCusker (Candidate)

ID: _____

Date: _____

TABLE OF CONTENTS

ABSTRACT

In this work the suitability of different methods of symmetric key distribution for application in wireless sensor networks are discussed. Each method is considered in terms of its security implications for the network. It is concluded that an asymmetric scheme is the optimum choice for key distribution. In particular, Identity-Based Cryptography (IBC) is proposed as the most suitable of the various asymmetric approaches. A protocol for key distribution using identity based Non-Interactive Key Distribution Scheme (NIKDS) and Identity-Based Signature (IBS) scheme is presented. The protocol is analysed on the ARM920T processor and measurements were taken for the run time and energy of its components parts. It was found that the Tate pairing component of the NIKDS consumes significants amounts of energy, and so it should be ported to hardware. An accelerator was implemented in 65nm Complementary Metal Oxide Silicon (CMOS) technology and area, timing and energy figures have been obtained for the design. Initial results indicate that a hardware implementation of IBC would meet the strict energy constraint of a wireless sensor network node.

LIST OF FIGURES

LIST OF TABLES

LIST OF ALGORITHMS

LIST OF ACRONYMS

ADC Analogue to Digital Converter

AES Advanced Encryption Standard

ALU Arithmetic logic Unit

APB Advanced Peripheral Bus

ASIC Application Specific Integrated Circuit

ASM Algorithmic State Machine

BDHP Bilinear Diffie Hellman Problem

CA Certificate Authority

CAD Computer Aided Design

CMOS Complementary Metal Oxide Silicon

COTS Commercial Off-The-Shelf

CPU Central Processing Unit

DHP Diffie Hellman Problem

DLP Discrete Logarithm Problem

DMA Direct Memory Access

DSP Digital Signal Processing

ECC Elliptic Curve Cryptosystems

ECDLP Elliptic Curve Discrete Logarithm Problem

ECDSA Elliptic Curve Digital Signature Algorithm

ECIES Elliptic Curve Integrated Encryption Scheme

EEA Extended Euclidean Algorithm

FFT Fast Fourier Transform

FPGA Field Programmable Gate Array

FSM finite state machine

GF Galois Field

GPS Global Positioning System

IBC Identity-Based Cryptography

IBE Identity-Based Encryption

IBS Identity-Based Signature

IBSC Identity-Based Signcryption

IC Integrated Circuit

IP Intellectual Property

ITS Intelligent Transportation Systems

Kbps Kilobits per second

KGC Key Generation Centre

LFSR Linear Feedback Shift Register

LOB line of bearing

LSB Least Significant Bit

Mbps Megabits per second

MEMS Micro Electro-Mechanical Systems

MSB Most Significant Bit

MTCMOS Multiple Threshold Complementary Metal Oxide Silicon

NIKDS Non-Interactive Key Distribution Scheme

NMOS n-channel Metal Oxide Silicon

OSI Open System Interconnect

PDA Personal Digital Assistant

PKI Public Key Infrastructure

PMOS p-channel Metal Oxide Silicon

RAM Random Access Memory

RF Radio Frequency

RFID Radio Frequency Identification

ROM Read Only Memory

RTL Register Transfer Level

SDF Standard Delay Format

SoC System on a Chip

SPEF Standard Parasitic Exchange Format

SRAM Static Random Access Memory

TSMC Taiwan Semiconductor Manufacturing Company

UART Universal Asynchronous Receiver Transmitter

VCD Value Change Dump

VHDL Very High Speed Integrated Circuit Hardware Description Language

WSN wireless sensor network

XML Extensible Markup Language

LIST OF PEER-REVIEWED PUBLICATIONS

B. Doyle, S. Bell, A. F. Smeaton, K. McCusker, and N. O'Connor., "Security considerations and key negotiation techniques for power constrained sensor networks," *The Computer Journal (Oxford University Press)*, vol. 49, no. 4, pp. 443–453, 2006.

K. McCusker, N. O'Connor, and D. Diamond., "Low-energy finite field arithmetic primitives for implementing security in wireless sensor networks," in *2006 International Conference on Communications, Circuits And Systems*, vol. III - Computer, Optical and Broadband; Communications; Computational Intelligence, June 2006, pp. 1537–1541.

K. McCusker, N. O'Connor, and D. Diamond, "A low-energy $GF(2^{4m})$ multiplier for security in wireless sensor networks," in *COOL Chips IX - IEEE Symposium on Low-Power and High-Speed Chips*, April 2006, pp. 279–290.

ACKNOWLEDGEMENTS

First of all, I would like to thank Rachel for her support over the years and her “gentle” encouragement to complete the thesis. I owe a great debt to my parents for instilling a love of learning in me that led me to undertake this work. My fellow group members, past and present, have been a great source of technical assistance and in particular I wish to acknowledge the help of Valentin Muresan, Andrew Kinane and Daniel Larkin. I greatly appreciate the help of Noel McCullagh in explaining the area of IBC to me, and also in casting a critical eye over this thesis. Finally, I would like to thank Noel O’Connor for his supervision and in particular his help in completing this thesis.

This thesis is dedicated to the best thing that has ever happened to me, my little daughter Éabha.

CHAPTER 1

Introduction

1.1 Wireless Sensor Networks: A new sensing paradigm

The doubling of transistors per unit area every eighteen months, according to Moore's law [68], has meant that the processing power of computers has increased greatly in the last few decades whilst their overall cost has decreased. This trend also makes it possible to provide on a die, which has area of the order of mm^2, enough computational power for a device to perform a meaningful function [45]. Low energy Complementary Metal Oxide Silicon (CMOS) radio [31] and Micro Electro-Mechanical Systems (MEMS) sensors, as demonstrated by Akustica [3], can also be manufactured using a standard CMOS process that also benefits from this shrinking of technology.

These developments mean that, in the near future, devices will be designed and built that have digital processing, a radio and a MEMS sensor incorporated on a single device or System on a Chip (SoC), known as a wireless sensor node, that is manufactured using a standard CMOS process. This device could potentially cost less than one dollar - Berkeley [45] has already demonstrated that this cost is possible for a SoC containing a Central Processing Unit (CPU), hardware accelerators and a radio. To date several platforms [44] have been proposed for devices that do not incorporate the sensor sub-system.

The sensors that can be deployed on these devices are transducers for detecting physical phenomena such as light, temperature, magnetic fields etc.. Also, chemical [42] and

biological sensors can be incorporated into the device using MEMS technology and lab-on-a-chip systems.

The small form factor, low cost, long lifetimes and wireless connectivity of the wireless sensor nodes opens the way to new applications for sensing technology, and also to much greater spatial density of sensors than can be achieved using conventional sensing technology. A network of these devices is known as a wireless sensor network (WSN). It could be envisaged in the future that wireless sensor nodes would be ubiquitous in our environment; interacting with end users and traditional networks in a seamless fashion; leading to a union between the physical world we inhabit and the digital world.

An overview of wireless sensors is given in several sources [76, 4, 22, 66, 45, 19, 30, 25]. The common characteristics that they all attribute to wireless sensor nodes are that they are cost very little and do not need their power source replaced or are self-powered with multi-year lifetimes typically envisaged on a single power source. The total energy in the power source for the wireless sensor node is typically considered to be of the order of 1000 J [45] and should have a power dissipation figure of 100 μW[76]. The network itself must be self-configuring and robust to individual wireless sensor node failure.

Application areas for WSNs, as can be seen in Section 1.2, are diverse. They could lead to significant advances in healthcare where patients could be continuously monitored in their own homes or in hospital without the need for health personnel to be physically present. Road safety and congestion reduction could be greatly helped by creating smart roads that have embedded wireless sensor nodes. Monitoring of the environment for harmful substances, such as from a chemical factory or a terrorist attack around important infrastructure, such as underground stations, could become practical.

This thesis is not directly concerned with these applications. Rather, the focus of this work is on making sensed data secure in practice - a key enabling technology for many applications.

1.2 Application Areas

Even though the computational and sensing ability of an individual node may be quite limited, the aggregated effect of all the sensors working together would be to provide a more accurate global picture of the spatial region in which the sensors are placed than could be achieved through conventional sensing technology. This opens up a whole vista of scenarios where sensor networks could be deployed that up until now could not be considered.

1.2.1 Intelligent Transportation Systems

Intelligent Transportation Systems (ITS) is the application of sensing, communication and control technologies to the road transportation system, with the aim of reducing congestion or improving road user's safety. It is a field in which wireless sensor networks could make a valuable contribution [21, 20]. One such approach is described in [20] and is outlined below.

Four sub-systems make up an ITS; the surveillance sub-system, strategy sub-system, execution sub-system and communication sub-system. Surveillance is currently carried out using ultrasonic sensors, video recognition and inductive loops. The communication sub-system uses a fixed wired network to send the data back to the strategy sub-system that decides upon the appropriate response. This response is then relayed to the execution sub-system that can change the traffic lights

In WSN ITS wireless sensor nodes are placed inside the vehicle, at fixed points along the side of the road and at intersections. The roadside nodes broadcast to the vehicle wireless sensor nodes and when they receive three such messages they can work out their position, which is sent back to the roadside devices. Of course, after this has happened for a second time the device in the vehicle can calculate its velocity. Its position and velocity is then sent to the nearest roadside device, which forwards this information onto the intersection wireless sensor node that interacts with the strategy sub-system to formulate an appropriate response, such as changing the lights.

The roadside device uses solar power but it still should consume as little power as

possible. The maximum power that it uses is $37mW$ and in the sleep state it consumes less than $1mW$. Security is essential in this system as it would be important that it could not manipulated to the advantage of an individual driver.

Another use of wireless sensor nodes in ITS would be in traffic speed enforcement. In the scheme above each vehicle could be issued with a wireless sensor node that would contain the registration details of the vehicle. In the Republic of Ireland every year new tax disks are sent out to an owner of a vehicle; this disk could be replaced with a wireless sensor node. If they are breaking the speed limit then this information could be forwarded by the wireless sensor network to a base station that sends the information to the authorities.

The cats' eyes on the road could be developed so that they incorporate wireless sensor nodes as well. If the devices cost $1 and assuming that the cats' eyes are $10m$ apart, then it would cost $100 per $1km$ of road to deploy this network. If these cats' eyes devices have sensors for detecting temperature, moisture etc. then they would be able to notify the drivers if there is an upcoming ice patch or other dangerous driving conditions.

In a city environment this wireless sensor network would provide a rich source of data for traffic managers. It could alert the managers that there is a major traffic jam on a certain route, and that a detour route is free of traffic. An alert could be issued to inform drivers travelling in that particular direction to avoid the congested route and take the detour. This could be introduced as a packet into the network and stored on wireless sensor nodes just before the detour.

The use of wireless sensor networks in ITS will enable much more data and different types of data to be incorporated into traffic management systems. There is also a need for security in the traffic enforcement scenario and also in the general case of information from wireless sensor nodes being able to contribute to the changing of traffic signals.

1.2.2 Healthcare

There are several projects in the application of wireless sensor nodes to the area of patient healthcare in emergency medicine [40, 34]. When the paramedics and doctors arrive at

mass casualty incident they have to classify the injured into critical (red), urgent (yellow) and minor (green). Following this initial classification they have to monitor the different patients to ensure that their condition does not deteriorate. If there are a lot of patients and not very many medical personnel this process would be time consuming and lead to the doctor having to stop treating a particular patient to check the vital signs of another patient. John Hopkins University [34] have designed a system, which uses the MicaZ mote from Crossbow Technology [24] to assist in this process.

Upon arriving at the scene the medical personnel attaches a sensor to the injured and presses a button on it to indicate the patient's status - red, yellow or green. This device can measure their heart rate, blood oxygenation level and blood pressure. It also uses Global Positioning System (GPS) for locating individuals. The medic interacts with the motes using a tablet PC and this can be a two way process. When treating another patient then the medic is kept informed, via the motes, of all the other patients in the area, thus resulting in better use of his time. There is also a capability for the wireless sensor node to connect to the Internet via the tablet PC. This enables the patient's records to be downloaded to the mote and also for hospitals accepting the casualties to have real time information on their condition.

Whilst taking the patient's pulse and oxygenation levels every second, reporting his/her position every five minutes and also taking blood pressure every fifteen minutes then the device operates for six hours, which in this case is an appropriate length of time. The data in the network is secured using an encrypted form of Extensible Markup Language (XML).

This type of system could also be used in a regular hospital scenario. When a patient enters hospital they could be issued with a device that has a wireless sensor node embedded in it. This wireless sensor node would be a repository for the patient's medical record, and during his time in hospital this would be continually updated as he receives treatment. Important information such as what drugs the patient should receive whilst in hospital could be held on the device. All drug containers could be attached with Radio Frequency Identification (RFID) tags that only have a range of less than a metre. When

the nurse is administrating the drug, the wireless sensor node could check to see whether the drug is correct. If it is not, then it sends a message packet to trigger a nearby alarm.

The wireless sensor nodes on the patients could have sensors for measuring heartbeat, temperature, etc. Any deviation from the norm for these metrics would be sent via other wireless sensor nodes in the ward back to the nurses' station. The nurses could also be alerted to movement of the patients as the their nodes interact with other nodes in the environment. This could be especially useful in a children's hospital.

In a healthcare application, security will be crucial to ensure the privacy of the patient's medical record. Also, it is not practical to replace batteries every six hours in a hospital situation so the devices should be designed to consume as little power as possible to ensure long life.

1.2.3 Environmental Monitoring

Sensors can play an important role in environmental threat detection. Chemical sensors attached to devices with integrated Radio Frequency (RF) transceivers can provide an early warning system to rescuers. They provide information on the toxic gas present and also the position of the contaminant.

The air quality is traditionally monitored by large sensors such as the AreaRAE IAQ from RAE [78]. The device measures $23.5cm \times 13cm \times 22.5cm$ and it weighs $3.9kg$. It has detectors for volatile organic compounds, temperature, humidity, carbon dioxide and toxic gases. Communication with a base station is achieved using a RF transceiver that has a range of two miles. It has a integrated GPS device. The device is powered either by a battery or connected to the mains. When powered by battery it has a lifetime of twenty-four hours. The price of this device is GB£4000.

The drawbacks of these traditional sensors are obvious. Due to their cost these devices could not be deployed in large numbers. This means that information that they transmit back to the base station will provide a very coarse picture on the position of the toxic gases. They will also have to be incorporated into the mains power supply to ensure that they operate over a long period of time; further increasing the cost of deployment.

When CMOS chemical sensors [42] are integrated to form wireless sensor nodes then this leads to a new approach to monitoring air quality. They could be deployed in large numbers, as they cost very little, and the cost of deployment of the WSN is negligible as they are not required to be linked to the mains power supply. The increased density of sensors leads to an accurate picture of spatial distribution of the toxic chemical and its concentration. This would help those charged with containment of the chemical to direct their attention to the source of discharge. Knowing the time and the place where the chemical was first detected will help the police to catch the perpetrators, as this information could be cross referenced with CCTV images to find out who was in the area at a particular time.

When this WSN is deployed in a counter-terrorism role it is important that the network is secure. The authorities will not want terrorist to gain control of the network to disable it or to mount a virtual attack, thus spreading panic.

1.3 Technical Challenges

From the discussion above it should be clear that there are formidable technical challenges to overcome before this vision of WSNs can become a reality, and some of these challenges are outlined below.

All of the envisaged applications require cheap sensors networks. The wireless sensor nodes themselves must cost very little, and this means that the devices must have a small silicon area to reduce their cost. Of course, this small die area means that memory and digital computational circuitry will be limited. This constraint places a burden on the chip designer implementing security on the device, as approaches that consume a lot of die area cannot be entertained.

In some of application areas of WSN it will not be possible, for reasons of cost or accessibility, to replace the power source when it is depleted. Conserving energy is vital for the network to operate as long as possible. The radio will be the main source of power dissipation in the device, and so should operate with a low duty cycle - less than 1%. It has been found that in a state of the art radio, the energy required to transmit one bit

can be dominated by the start-up energy. This is because the packets to be transmitted are typically very small in wireless sensor networks. The transceiver should be designed so that the start-up time before the data can be transmitted or received is as small as possible [92]. The digital circuitry should be designed with the aim of reducing the energy consumption using well-known techniques (see Section 5.2.1).

It is also important that establishing the network is not an expensive process. It should be possible for a person who is not an engineer to deploy these networks. Therefore the WSN has to be self-configuring, and robust to individual device failure. Of course, the application programming of the wireless sensor nodes would have been carried out by an engineer but the end user should just be able to scatter the wireless sensor nodes and expect them to autonomously establish a viable WSN.

A measurement taken from a wireless sensor node consists of three main components; the physical measurement, the time it was taken and the position of the device. Synchronisation of the devices will be required to get a valid time stamp for the reading. This can be achieved by the broadcasting of synchronisation packets from a wireless sensor node within the WSN as there may only be occasional interaction with the gateway device that injects or extracts data from the WSN. The end users of the system will not be concerned with a reading from an individual node but rather with the position from which the reading originated. Therefore it is also required that the wireless sensor nodes know their position, at least relative to one another, and this is a challenging problem (see Section 2.4).

Most of the applications outlined in this thesis (see Section 1.2) require some level of security: the driver who is speeding will not want this information to become public; the company that is measuring the pollution that they are creating in the environment will wish to release that information in a controlled way; the patient in the hospital will want to be sure that his private medical records remain private. So, in order for these networks to be deployed in most applications, it is essential that the issue of security is solved, as noted in [74, 18]. As will be discussed in this thesis, a symmetric key cryptosystem is appropriate for communication between the wireless sensor nodes in a WSN because

it has minimal message expansion and so does not greatly increase the number of bits that have to be transmitted (see Chapter 4). It is also not costly in term of computational power and hence digital energy dissipation. How the symmetric keys are deployed in the network however is a challenging problem, and is the main focus of this thesis.

1.4 Research Objectives

The objectives of this research are as follows:

1. Investigate techniques for implementing security in a wireless sensor network. The approaches should be evaluated in terms of their practicality and energy dissipation. This investigation will take the form of a technical review of known techniques.

2. Propose a protocol to distribute symmetric keys in the network that could solve the key distribution problem. Evaluate this protocol in terms of the time and the energy required to implement it in software. This evaluation will involve running the software on an ARM platform [5] and measuring the energy consumption.

3. Using the figures from the evaluation of the protocol, develop a hardware architecture for the most computationally demanding element of this protocol and implement this using a standard cell $65nm$ library.

4. Validate this work against the software approach in terms of energy and time. The work should also, as much as is feasible, be validated against the prior art in the research space.

1.5 Thesis Structure

The remainder of this thesis is structured as follows:

- *Chapter 2 – Technical Background*
  Introduces the devices that are used to establish a WSN and the technical challenges facing this emerging technology.

- *Chapter 3 – Mathematical Background*

  The mathematics of Galois field algebra and the Tate pairing are described in detail to enable a better understanding of later chapters.

- *Chapter 4 – Security*

  The suitability of different methods of key distribution for application in wireless sensor networks are discussed. Each method is considered in terms of its security implications for a wireless sensor network. Identity-Based Cryptography (IBC) is proposed as the most suitable candidate for secure distribution of keys in the network.

- *Chapter 5 – Arithmetic Units*

  The Galois field arithmetic units that make up the Tate pairing are discussed and implemented in hardware. Timing and energy figures for the various units are also presented.

- *Chapter 6 – Tate Pairing Accelerator*

  The overall architecture for the Tate pairing is presented and implemented. Timing and energy figures for the Tate pairing are presented. These are also compared against previous implementations and software.

- *Chapter 7 – Conclusions & Future Work*

  A summary of thesis contributions and research objectives attained are presented, as are plans for future work.

- *Appendix A – Algorithmic State Machines for the designs*

  The Algorithmic State Machine (ASM) diagrams of the circuits discussed in Chapter 5 and Chapter 6 are presented.

- *Appendix B – Programmer's Model*

  A description of the registers to aid programming of the module.

- *Appendix C – Signal Descriptions*

  The signals that interface with the module are described.

1.6 Summary

WSNs were introduced in this chapter. Due to their small size, low cost, long lifetime and wireless connectivity, they could enable a new paradigm of ubiquitous sensing to become a reality. Barriers between the physical world and digital world could be broken down and people and machine could interact with the physical world in ways not previously possible.

The fields in which WSNs might find application are varied and examples are; ITS, healthcare and environmental monitoring. All of the example applications discussed in this chapter will need secure networks, though the environmental monitoring example will need the highest level of security when it is deployed in a counter-terrorism role. What also differentiates this WSN from the other examples mentioned is that the wireless sensor nodes have a fixed position and this fact can help secure the network (see Chapter 4). This will be the target application for this work.

CHAPTER 2

Technical Background

2.1 Introduction

This chapter concerns itself with a more in depth look at the technical requirements of the wireless sensor nodes, and the challenges faced when deploying a WSN.

2.2 Wireless Sensor Node Platforms

There are many different platforms for implementing a WSN [44] but they all share common characteristics. Figure 2.1 shows a generic wireless sensor node. The key subsystems are the sensor, radio and controller. The sensors that are to be integrated with the device will obviously be application dependent. There has been progress recently in integrating a MEMS sensor on CMOS using a standard process [3]. To date no group has integrated all of these key subsystems onto a single die. Communication between wireless sensor nodes is achieved using a radio signal. The radio should be designed to have a small start-up time (see Section 2.5.2) and operate using a low duty cycle. The digital computation required to run the radio could be implemented in a baseband processor or by the general purpose processor [43]. This processor also has to control the sensor and read back values from the Analogue to Digital Converter (ADC). Read Only Memory (ROM) is required for storing the operating system and applications that are run by the processor.

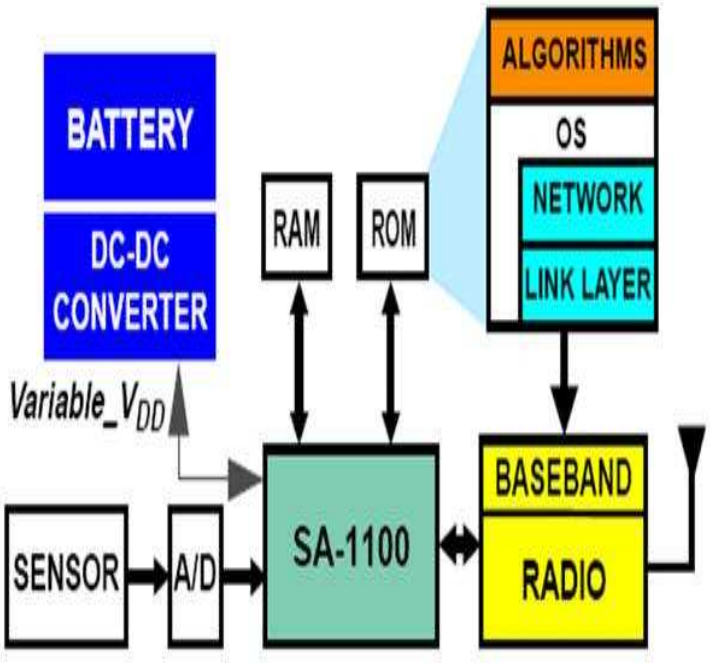

Figure 2.1: Architectural overview of a wireless sensor node [19]. The main blocks are the sensing circuitry, the processor (SA-1100) and the radio.

It also stores the firmware for running the radio baseband processor. As the devices will spend most of lifetime turned off [45] they must store sensor readings in their persistent memory as well. Random Access Memory (RAM) will be needed for program execution. Energy will generally be provided to the wireless sensor node by way of a battery. If the processor has to the ability to operate at different voltage levels then a DC-DC converter could be added to the architecture to enable variation of the voltage.

There are many research groups [43, 9, 67] and companies [2, 73] working on hardware implementations of wireless sensor nodes and below is a description of some of the devices that have gained prominence in the research field.

2.2.1 Mica

The Berkeley Mica architecture is presented in [43] and it is a Commercial Off-The-Shelf (COTS) solution. It consumes $45mW$ in its active state and $30\mu W$ in its sleep state.

The core of the design is an Atmel ATmega128L 8-bit micro-controller, which has standard digital and analogue interfaces. There is also a radio transceiver, a co-processor

with external memory to handle wireless reprogramming, and a DC-DC converter to provide a constant 3 Volts. It features $128Kb$ of instruction memory and $4Kb$ of data memory. There is also a $512Kb$ flash memory for measurement storage.

It does not have a separate processor for application and protocol processing but relies on an event based operating system and hardware accelerators to provide sufficient performance. The timing hardware accelerator catches the edge transition of the timing pulse. Once the timing information is captured, software then uses it to configure a serialization accelerator that automatically times and samples the individual bits. In receive mode, this accelerator provides a byte to the main datapath. In transmit mode, the micro-controller writes a byte at a time to this accelerator, which in effect acts like a type of Direct Memory Access (DMA). It can operate up to 100 Kilobits per second (Kbps). Security can be implemented in software, on these devices, as there is no crypto accelerator.

2.2.2 Spec

Berkeley [45] have implemented a wireless sensor node minus the sensor system on an Application Specific Integrated Circuit (ASIC) using a $25\mu m$ process. It measures $2.5mm \times 2.5mm$, and it consumes $27mW$ in its active state and $2\mu W$ in its sleep state. An architecture of the device is shown in Figure 2.2. It is estimated that this device would cost less than one dollar if manufactured in sufficient quantities.

It uses an 8-bit RISC CPU that has 16-bit instructions. It can operate at a frequency of 4Mhz. There is $3Kb$ of Static Random Access Memory (SRAM) organised into 512 bytes pages that can be mapped to the data or instruction memory by the address translation unit; this is so as to enable field programming of the device.

There is no baseband processor but computationally intensive operations of the RF communications, such as synchronisation and serialisation, are implemented as hardware accelerators. These accelerators can be used by the CPU to implement the radio protocol. There is no receiver in this device. It has a transmission rate of 50 Kbps.

Other components that are present are; the ADC for connecting to the sensor data, the analogue part of the RF sub-system, GPIO ports, system timers and an encryption

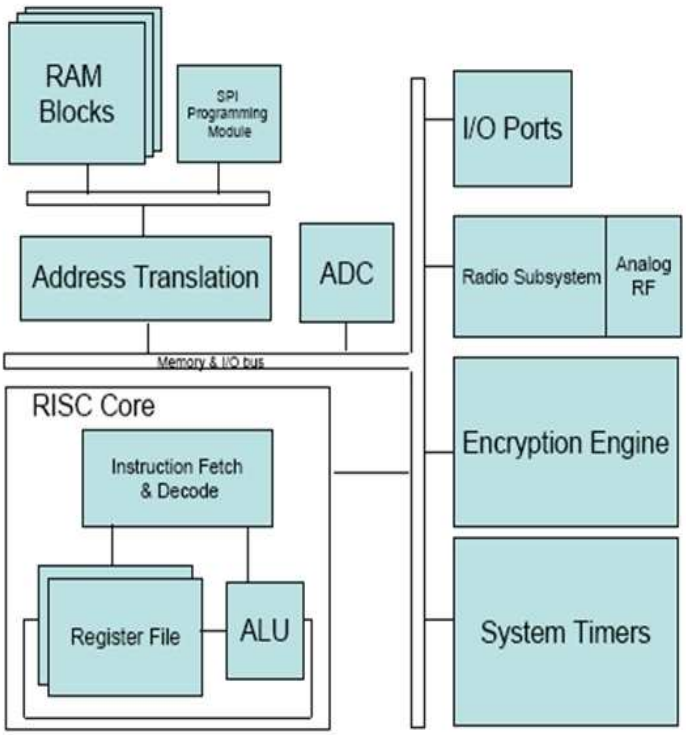

Figure 2.2: Architectural of Spec [45]

accelerator. Encryption is achieved using four 40-bit linear feedback shift registers that are XORed together to obtain a pseudo-random sequence. They are seeded by software and two devices would have to agree on this seed value before being able to communicate securely.

2.2.3 Intel Mote 2

Intel have developed a more powerful platform [2] that is designed for use with more intensive Digital Signal Processing (DSP) type operations, such as on video or acoustic data. It is a COTS implementation. In its active state it consumes less than $0.21W$ and in its deep sleep state it consumes $331\mu W$.

It is designed around a XScale processor [48](PXA27x), which is a 32-bit device, and its operating frequency is in the range of 13 MHz to 416 MHz. It has a minimum of $512Kb$ of SRAM that can be augmented with a maximum of $32Mb$ of SDRAM. Flash memory in the device ranges from $32Mb$ to $64Mb$. For communication it uses a IEEE 802.15.4 (Zigbee) transceiver that can operate at 250 Kbps. The CPU has the standard

I/O interfaces integrated, such as I^2C, SSP, UART, GPIO etc., but it also interfaces to audio and camera devices. There is an accelerator for multimedia operations. Security is implemented by using the Intel Wireless Trusted Platform which provides cryptographic engines for a full range of security primitives such as, random number generation, symmetric and asymmetric cryptography, key exchange etc.

.

2.2.4 Conclusions

There is no one application scenario for WSN [22] and as such, the specification requirements of the wireless sensor nodes will differ depending on their purpose. It has been argued [44] that the devices fall into four broad categories; specialised for low bandwidth sensing, generic low bandwidth sensing and communication relay, high bandwidth sensing, and a gateway device for connection to traditional networks. Spec is an example of a specialised device, Mica is a generic device and Intel mote 2 is used for high bandwidth sensing.

Energy is a key design constraint for all these categories of devices. Cost will be a bigger issue with specialised devices as they will be deployed in great numbers. This thesis targets the area of low bandwidth sensing devices that will have a fixed mission at deployment, similar to the Spec device.

Only the high bandwidth device, Intel Mote 2, has dedicated hardware support for asymmetric cryptography. The devices that are being targeted in this research either have no crypto engines, or rely on software to provide secure data transmission. They do not have accelerators for the asymmetric cryptosystem that is required to solve the key distribution problem in WSN. A key goal of this research to investigate a solution to this problem in an energy constrained device, and to verify and validate the proposed solution.

2.3 Network Establishment

The wireless sensor nodes must set up an ad-hoc network in order to send data back to the base station that extracts data from the network. How an individual wireless sensor node sends its data back to the base station depends on whether the base station is within range of the wireless sensor nodes, which is the scenario considered in [100]. If the wireless sensor nodes are capable of changing their transmission power output then there are three methods of data transfer.

1. Every wireless sensor node transfers its data independently to the base station.

2. Multi-hop networking. Intermediate wireless sensor nodes on the way to the base station act as relays for the data. This can reduce energy, as the path loss term is proportional to distance squared. If the energy required by the transmitter in order for the receiver one metre away to receive the signal is $1nJ$ per bit, then the energy required to send this data 50m will be $50{\times}50{\times}1$ or $2500nJ$, but for five hops of 10m it will be $5 \times 10 \times 10 \times 1$ or $500nJ$. Hence there is an eighty per cent improvement in energy efficiency using a multi hop method.

3. Cluster protocol. In this case, wireless sensor nodes in the same area have collaborative data and so these elect a clusterhead that fuses the data before transferring it to the base station. The clusterhead has to do extra computation and also has an increased communication cost, because of this the clusters and clusterheads are changed dynamically.

The method used depends on the number of wireless sensor nodes and distance from the base station. It has been shown [100] that for distances greater than twenty metres the cluster protocol is the most efficient.

The discussion above assumes that the clusterhead wireless sensor node can transmit to the base station. When the wireless sensor node's transmission range is limited, and the network is spread over a large geographical area where the individual wireless sensor nodes do not have access to the base station then a multi-hop networking protocol is required (see Figure 2.3). In this scenario the wireless sensor nodes in a particular region

collaborate to send their data to a clusterhead wireless sensor node that performs some DSP on the data before forwarding it in a multi-hop fashion to the base station.

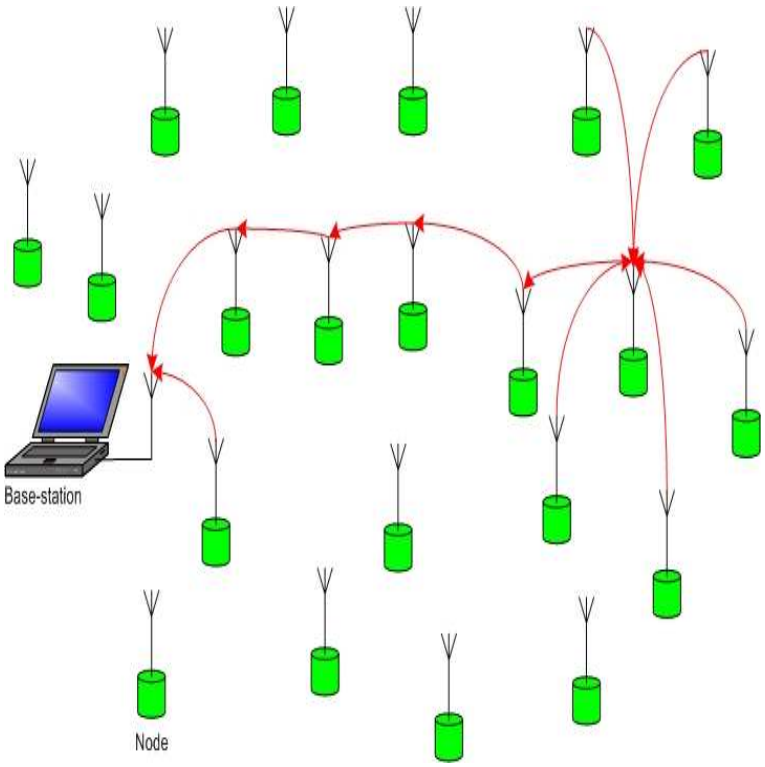

Figure 2.3: Multi-hop network

In general the routing protocols can be divided into proactive and reactive routing paradigms [88]. For a proactive protocol, all the wireless sensor nodes store a routing table that enables them to communicate with every other wireless sensor node in the system. Any change in the topology would have to be communicated to every wireless sensor node so that it can update its routing table. In the case of reactive routing the network only finds routes as they are required. This approach is more suited to distributed wireless sensor networks, as it lends itself to low duty cycle radio transmission. Reactive routing can be either destination-initiated, where the data consumer (i.e. the base station) begins the route discovery process, or source-initiated, which is the opposite.

One such reactive routing scheme is directed diffusion [47]. This is a data centric protocol that uses a destination-initiated routing scheme. The scheme does not request data from a particular wireless sensor node but from a geographical area, hence a global unique addressing system is not required. The sink wireless sensor node (i.e. base station)

might ask a question such as, “what is the pH in a region X”. This request for data is known as an interest, and it takes the form of an attribute-value pairs, which in this case could be the pH level, geographical area and a data rate. The scheme has an exploratory phase and a reinforcement phase. In the first phase the data query is diffused (broadcast) through the network with a much reduced data rate in order to evaluate if the interest can be met. This flooding results in gradients being set up that specify the direction of the wireless sensor node from where the interest came from. If this first phase is successful then the data originator can reinforce a path that meets the user criteria of low latency and low energy. This reinforcement takes the form of increasing the data rate from the data source to the data sink. There can also be in-network data aggregation to improve the quality of the result.

In this thesis we assume that a network has been established. The form that the network takes is a multi-hop network where wireless sensor nodes can collaborate in cluster type formations before sending data back to the base station (see Figure 2.3). We further assume that a routing algorithm has been implemented that ensures a robust routing channel between two nodes in the network. Network establishment and routing protocols in wireless sensor networks are challenging and active areas of research but will not be considered further in this thesis.

2.4 Localisation

In order for the sensing measurements to be meaningful, and in order to derive the topology of the network, it is necessary for the wireless sensor nodes to know their global position. One method to do this would be to use GPS, but this would be a costly option in monetary terms. Another method is to use the radio signal to work out the distance of each wireless sensor node from some pre-determined points [87] called anchor wireless sensor nodes, and then triangulation can be used to work out each wireless sensor node’s global position.

A two-staged algorithm is proposed in [87]. Each of the wireless sensor nodes propagates a relative positional algorithm through the network. When the wireless sensor node

is in the relative positional map of four or more anchor wireless sensor nodes, it can compute its distance to each anchor device, and then through a process of triangulation it can work out its own position. It then enters the second stage and performs an iterative process of triangulation with its adjacent wireless sensor nodes until they agree on their position. A variation on this method is also proposed [86] that counts the number of hops and uses the range of each wireless sensor node to work out the distance, instead of calculating the absolute distance to the anchor devices. It is assumed that in this thesis that the wireless sensor nodes can run a localisation algorithm in order to determine their relative position using the techniques outlined above.

2.5 Energy Dissipation

Perhaps the most onerous challenge facing the emerging technology of wireless sensor networks is the requirement that the devices will have to operate for long periods in hostile or inaccessible environments where the replacement of the power source would not be possible. It is of course, for reasons of cost, impractical to replace the power source in a network containing hundreds or thousands of wireless sensor nodes even if they are accessible. For this reason it is imperative that the system architecture of the wireless sensor nodes and the network as a whole should be designed with the aim to minimising energy dissipation.

There are three sources of energy dissipation in a wireless sensor node; sensing circuitry, digital computation and radio circuitry (see Figure 2.1). The sensing circuitry consists of the actual sensor itself and all the amplification and filtering components that are required. Digital computation is used to provide the DSP of the sensor data and for implementing the Open System Interconnect (OSI) protocol stack. Finally, the radio circuitry consists of analogue electronics that are responsible for modulating / demodulating and actual transmission of the radio signal. This thesis is primarily concerned with minimisation of digital computational energy.

2.5.1 Digital Computational Energy Dissipation

At the device level there are several methods of reducing computational energy consumption. The most common approach is to migrate computationally intensive algorithms from software to hardware [97]. The software energy profile itself can be modelled [94] that results in the following equation for the total energy used by a program;

$$E_{total}(V_{DD}, f) = C_{total}V_{DD}^2 + (N/f)V_{DD}I_O e^{\frac{V_{DD}}{nV_t}} \tag{2.1}$$

where C_{total} is the total capacitance switched by the program, V_{DD} is the operating voltage, N is the number of cycles the program takes to execute, f is the frequency of operation, I_O and n are processor dependent variables, and V_t is the threshold voltage. From this equation it can be seen that a lowering of the voltage leads to a marked reduction in energy consumption. When the number of processes running on a processor reduces but the latency remains the same, then the frequency and hence the voltage can be reduced. This technique is known as dynamic voltage scaling [38]..

Another technique is to use an ensemble of systems [10]. An example of this is a multiplier circuit that has a worst case operating condition of a 32×32 multiplication, but this does not occur often, and usually there is a 8×8 or 16×16 multiplication. This would mean in the 8×8 case that there would be switching on all 32 bit width of the circuitry even though an 8 bit wide operand was required. This could be ameliorated by having three separate multiplication circuits and some overhead to decide which one is suited for a particular operation.

Another method that the wireless sensor node can use to make energy saving is dynamic power management [95]. In this scheme the wireless sensor node enters a lower power state when it is not required to be in active mode. The lowest state it could reach would be for the sensing and RF circuitry to be turned off, and the processor and memory to be in sleep mode. It has been shown in [95] that there are several states that a system can enter, and that in general the lower the state in terms of energy, then the longer it takes to go into and out of that state, and also the greater the energy overhead in making the

transition. An analytical formula is presented for the minimum time, $T_{th,k}$, required for the transition to be worthwhile;

$$T_{th,k} = \frac{1}{2}[\tau_{d,k} + \left(\frac{P_0 + P_k}{P_0 - P_k}\right)\tau_{u,k}] \tag{2.2}$$

where P_0 is the higher and P_k is the lower energy level, $\tau_{d,k}$ is the transition time form P_0 to P_k and $\tau_{u,k}$ is the transition time back to P_0. As long as the time between events that the wireless sensor node needs to process is greater than this minimum time then the wireless sensor node may enter a lower energy level.

In wireless sensor node the higher layers should be able to adapt the hardware in response to changes in the system state, or user quality requirements [19]. For example, in order to extend the lifetime of the device the user may request less sampling or a lower duty cycle radio transmission. Furthermore, the algorithms that are run on the data could be curtailed. These last two examples would lead to a decrease in quality of the result but also a reduction in energy usage. This trade off between energy and quality is the theme of several papers [10, 96, 41].

It can be shown [96] that minor changes to the algorithms used can lead to significant improvement in the Energy-Quality behaviour. A simple example of this would be a power series with ten terms. If the value is less than one, then the first few terms make the biggest contribution to the overall result. If the value is greater than one then the final few terms are the most important. Thus the software / hardware can be adapted to first check if the value is greater than one, and then proceed with the calculation until the required accuracy is obtained. This method is proven for several common DSP algorithms [96].

In this work computationally intensive operations in software are identified and migrated to hardware. Dynamic voltage scaling is not implemented as we are working at the block level but it is included in our overall architecture discussion. It is assumed that the energy and cost of the block are the main design constraints for implementing the security protocol, but overall energy of the device must also be considered so the performance of the block must be such that the device can enter a low power state as fast as possible. The question of quality / energy trade off for security is not dealt with in this work as a fixed

application is assumed.

2.5.2 Radio Energy Dissipation

The radio transceiver should be designed and operated so as to consume as little energy as possible. This necessitates a low duty cycle operation. The energy required to transmit an n-bit packet is given by Equation (2.3) [65],

$$E_{tx}(n, R_{code}, P_{amp}) = T_{start}P_{start} + \frac{n}{RR_{code}}(P_{txElec} + P_{amp}) \tag{2.3}$$

where P_{start} and T_{start} are power and time required for starting up the transceiver, R is the bit rate, R_{code} is the coding rate, P_{txElec} is the digital power consumed during transmission and P_{amp} is the power required by the amplifier, and it is a function of the transmission power. The energy required to receive a n-bit packet is given in Equation (2.4) [65],

$$E_{rcvd} = T_{start}P_{start} + \frac{n}{RR_{code}}P_{rxElec} \tag{2.4}$$

where P_{rxElec} is the energy required to run the radio front-end. The radio should operate on a low duty cycle [45] (less than 1%) in order to conserve energy. It can be seen from Equations (2.3) and (2.4) that the packet size, n, should be kept as small as possible.

The transceiver should be designed so that the start-up time before the data can be transmitted or received is as small as possible [92]. This is because the packets to be transmitted are typically very small in wireless sensor networks and the start-up energy can be a significant component of overall energy transmission costs. A low power $0.25\mu m$ CMOS transmitter has been designed [31] that has a start-up energy cost of $440nJ$ and costs $10.8nJ$ to transmit a bit at 2.5 Megabits per second (Mbps).

It has been suggested in [76] that if the wireless sensor node's power requirement is of the order of $100mW$ then they could scavenge their energy source from the environment. This approach has been used in [82] to develop a self-contained radio transmitter, which harvests its energy from the environment, using vibration or solar energy sources. It has been shown that this approach can extract up to $180\mu W$ under optimum conditions. In

this thesis it is assumed that there is a fixed energy reserve.

2.5.3 Sensor Energy Dissipation

MEMSs sensors can be integrated onto the same die as the radio and digital electronics and so they will consume less energy than a separate chip solution. The sensor on the device will consume energy in the region of milliwatts. A low power CMOS MEMS motion sensor has been demonstrated that consumes $15mW$ of power [84]. In the case of Akustica [3] a MEMS microphone with a max power figure of $2.7mW$ is possible. As with the radio sub-system, it is important that these devices operate on a low duty cycle. Therefore, when designing a wireless sensor node device it is imperative that all the computation for the sub-systems is performed as fast and possible so that they can then enter their low energy state.

2.5.4 Network level Energy Dissipation

The parallel processing capability of the network can be leveraged through distributed processing to reduce energy dissipation. In [99] an acoustic application to try and find the line of bearing (LOB) to a source is evaluated. One sensor receives data from all the other wireless sensor nodes in its vicinity. It must then perform a beam-forming algorithm that fuses multiple streams of correlated data input into a single high quality output. It then transmits this data to the base station. This particular algorithm takes two stages the first of which is a Fast Fourier Transform (FFT) on the sensors data. The FFT data is then beam-formed to find the LOB of the source. The FFT operation can be done sequentially on the wireless sensor node doing the beam-forming or in parallel on the wireless sensor nodes doing the sensing. In the parallel case, and if the latency does not change, then dynamic voltage scaling can be used to reduce the energy dissipation of the algorithm as a whole. It has been shown that the distributed technique has led to a 45% improvement in energy dissipation. This example shows that, as well as considering the energy requirements of the individual wireless sensor nodes, the total energy dissipated in the network should also be considered [79].

2.6 Security

In order for these networks to be deployed in the field it is essential that the issue of security is solved as noted in [74, 18]. The challenge for implementing security in WSNs is that there is not a predetermined infrastructure with which the wireless sensor nodes can interact. This makes it difficult to distribute symmetric keys to the node using a traditional approach. This work proposes Identity-Based Cryptography (IBC) as a solution to the key distribution problem.

2.7 Summary

The design challenges and issues facing the emerging technology of WSNs are that the devices making up the network must cost very little, operate for a long time on a single power source, self-configure themselves into a robust self healing network, determine their location relative to each other and secure the network against interception of data.

In order for WSN to be deployed in the field it is essential that the issue of security be solved [74, 18]. The challenge for implementing security in WSNs is that there is not a predetermined infrastructure with which the wireless sensor nodes can interact. This makes it difficult to distribute symmetric keys to the node using a traditional approach. This work proposes Identity-Based Encryption (IBE) as a solution to the key distribution problem. Security will be discussed in greater depth in Chapter 4 after its mathematical background is introduced in the following chapter.

CHAPTER 3
Mathematical Background

3.1 Introduction

This work is primarily interested in applying IBC to solve the key distribution problem in WSN. The IBC protocol considered uses the Tate pairing, which is a bilinear pairing defined over an elliptic curve, that takes its coordinates from a finite field. Although the focus of this thesis is not the underlying mathematical framework, knowledge of abstract algebra, elliptic curves and the Tate pairing is required in order to understand later chapters in this thesis. For this reason, these topics are covered in detail here.

3.2 Groups, Rings and Fields

In order to aid understanding of later chapters of this thesis the abstract algebra concepts of Groups, Rings and Fields are introduced. Proofs and further information regarding these topics can be found in references [62, 103].

3.2.1 Group

Definition 3.1 (Group). *A group $\mathbb{G}$ is set of elements upon which binary operator "$\times$" has been defined. The operation has the following properties.*

1. Closure: $a \times b \in \mathbb{G}, \quad \forall\, a, b \in \mathbb{G}$

2. Associativity: $(a \times b) \times c = a \times (b \times c), \quad \forall\, a, b, c \in \mathbb{G}$

3. Identity: $\exists\, e \in \mathbb{G}, \quad a \times e = e \times a = a, \quad \forall\, a \in \mathbb{G}$

4. Inverse: $\exists\, a^{-1} \in \mathbb{G}, \quad a \times a^{-1} = a^{-1} \times a = e, \quad \forall\, a \in \mathbb{G}$

If the group satisfies the condition below then it is known as an Abelian group

5. Commutativity: $a \times b = b \times a, \quad \forall\, a, b \in \mathbb{G}$

When a group has an operation that is similar to addition it is known as an additive group and the operation is represented by "$+$". In this case "0" is the identity element, and the inverse of "a" is given by "$-a$". A multiplicative group has an operation that resembles multiplication and its operator and inverse are represented as in the list above. In this case "1" is used for the identity element.

The order of a group is given by the cardinality of the set on which the group is defined. An example of an infinite Abelian group is the set of integers under addition. Cryptography is primarily interested in finite groups. Modulo addition can be used to define a finite group over the set of integers such that $x + y \equiv z \pmod{n}$ where x, y, z and n are integers. This leads to the integers being partitioned into n equivalence classes that can be labeled $\{0, 1, 2 \ldots n-1\}$. The order of an element, α, of a finite multiplicative group is defined as the smallest integer β such that $\alpha^{\beta} = 1$.

3.2.2 Ring

Definition 3.2 (Ring). *A ring $\mathbb{R}$ is set of elements upon which binary operations "$\times$" and "$+$" have been defined. The following properties are also required.*

1. $\mathbb{R}$ forms an Abelian group under "$+$"

2. Closure under "$\times$": $a \times b \in \mathbb{G}, \quad \forall\, a, b \in \mathbb{G}$

3. Associativity under "$\times$": $(a \times b) \times c = a \times (b \times c), \quad \forall\, a, b, c \in \mathbb{R}$

4. Distributivity of "$\times$" over "$+$": $a \times (b + c) = (a \times b) + (a \times c), \quad \forall\, a, b, c \in \mathbb{R}$

If the ring satisfies the condition below then it is a commutative ring

5. Commutativity under "$\times$": $a \times b = b \times a, \quad \forall\, a, b \in \mathbb{G}$

If the ring satisfies the following condition then it is a ring with identity

6. Identity: $\exists\, e \in \mathbb{R}, \quad a \times e = e \times a = a, \quad \forall\, a \in \mathbb{R}$

The integers are an example of a ring where the operations are addition and multiplication. Another example of a ring is the set of polynomials with binary coefficients and it is denoted as $\mathbb{F}_2[x]$

3.2.3 Field

Definition 3.3 (Field). *A field $\mathbb{F}$ is set of elements upon which a binary operations "$\times$" and "$+$" have been defined.The following properties are also required.*

1. $\mathbb{F}$ forms an Abelian group under "$+$"

2. $\mathbb{F} - \{0\}$ forms an Abelian group under "$\times$"

3. Distributivity of "$\times$" over "$+$": $a \times (b + c) = (a \times b) + (a \times c), \quad \forall\, a, b, c \in \mathbb{F}$

The set of real numbers form a field, whereas the integers do not, as every element does not have a multiplicative inverse. A finite or Galois Field (GF) can be generated from the ring of integers using modulo p addition and multiplication, where p is a prime number. The resulting GF has elements $\{0, 1, 2 \ldots p - 1\}$ and is known as a prime field. It is represented by the symbol $GF(p)$, or $\mathbb{F}_p$.

The field $GF(2)$ contains two elements $\{0, 1\}$ and its addition operation has the same functionality as a logical XOR gate. Also the multiplication operation can be represented by the logical AND gate.

3.2.4 Galois Fields of order p^m

Analogously to the integers, a finite field can be defined over the ring of polynomials $GF(p)[x]$ where p is a prime number. The field is constructed by performing modulo $f(x)$ reduction of the ring $GF(p)[x]$, where $f(x)$ is an irreducible polynomial of degree m with coefficients $\in GF(p)$, such that it has a root α of order $2^m - 1$. $f(x)$ and α are known as a primitive polynomial and primitive element respectively of the field $GF(p^m)$.

All finite fields have p^n elements, where n is an integer and are unique up to isomorphism. The field $GF(p^m)$ can be considered as an extension field of the subfield $GF(p)$. An extension field of $GF(p^m)$ can be constructed by using the same technique as above but now the primitive polynomial will have coefficients $\in GF(p^m)$ and have degree k. This results in the finite field $GF(p^{mk})$. In this thesis, only finite fields of the form $GF(2^m)$ and $GF(2^{mk})$ are considered as they map easily to hardware.

The characteristic of a field is the smallest integer, *char*, such that when an element from the finite field is added *char* times then the result is "0". In the case of $GF(p^m)$ the characteristic is p as all addition is performed modulo p.

3.2.5 Basis Representation

A finite field is a vector space and as such, can be spanned by a set of elements. For $GF(p^m)$, it has a basis of dimension $m - 1$. Thus all elements of $GF(p^m)$ can be written as a linear combination of the basis vectors

$$\gamma = \{a_0\beta_0 + a_1\beta_1 + a_2\beta_2 + a_{m-1}\beta_{m-1}\}$$

where $a_j \in GF(p)$, and $\beta_j \in GF(p^m)$ are the basis vectors. There are a number of different examples of basis such as polynomial or normal. For this work the basis that has been selected is polynomial. The squaring operation and square root operation are easier to implement in normal basis but multiplication is much more costly in terms of area and performance than if these arithmetic operations were implemented in a polynomial basis. And as is shown later, there are more multiplications than squaring or square root

operations required to implement the Tate pairing. Due to the dominance of multiplication over the squaring and the square root operation it has been found that in characteristic three an implementation of the Tate pairing using a polynomial basis performs better than one using a normal basis [36]. The assumption for this work is that this result would also hold in characteristic two.

Definition 3.4 (Polynomial Basis). *The polynomial basis is a set of elements* $\{\alpha^0, \alpha^1, \alpha^2 \ldots \alpha^{m-1}\}$ *where* $\alpha \in GF(p^m)$ *is the root of the primitive polynomial* $f(x)$ *used to define the field* $GF(p^m)$.

Elements of the GF can be represented using a polynomial $GF(p)[\alpha]$ of degree $m-1$ i.e. $\{a_{m-1}\alpha^{m-1} + a_{m-2}\alpha^{m-2} \ldots a_1\alpha^1 + a_0\}$ where $a_{0\rightarrow(m-1)} \in GF(p)$. The multiplicative structure of the finite field can also be represented as a series of powers of α i.e. $\{\alpha^0, \alpha^1, \alpha^2 \ldots \alpha^{2^m-2}\}$.

3.2.6 Hardware Representation

The polynomial $f(x) = x^3 + x + 1 \in F_2[x]$ is irreducible as $f(1) = 1$ and $f(0) = 1$, and therefore it has no factors in the field $GF(2)$. It can be used to generate the finite field $GF(2^3)$. Every polynomial $g(x) \in F_2[x] \pmod{f(x)}$ is equivalent to a polynomial $R(x)$ that is of order less than three, and $R(x)$ is represented as below.

$$R(x) = r_2x^2 + r_1x + r_0 \pmod{f(x)}, \quad \forall r_j \in GF(2). \tag{3.1}$$

In order to represent this element $R(X)$ in hardware three D-type flip flops are required, where the Most Significant Bit (MSB) represents r_2. When a element such as $R(x)$ is multiplied by x this results in a polynomial that has a term in x^3 which must be reduced modulo $f(x)$.

$$\begin{aligned} xR(x) &= r_2x^3 + r_1x^2 + r_0x \pmod{f(x)} \\ &= r_2(x+1) + r_1x^2 + r_0x \pmod{f(x)} \\ &= r_1x^2 + (r_0 + r_2)x + r_2 \pmod{f(x)} \end{aligned} \tag{3.2}$$

Multiplying by x is equivalent to a shift in a shift register with the MSB connected to the Least Significant Bit (LSB), and the value of the register representing the x term is generated from adding r_0 and r_2 in $GF(2)$. The x term is generated in hardware by $r_0 \oplus r_2$, where $\oplus$ is the symbol for an XOR gate. Thus, multiplying by x requires a Linear Feedback Shift Register (LFSR) as in Figure 3.1.

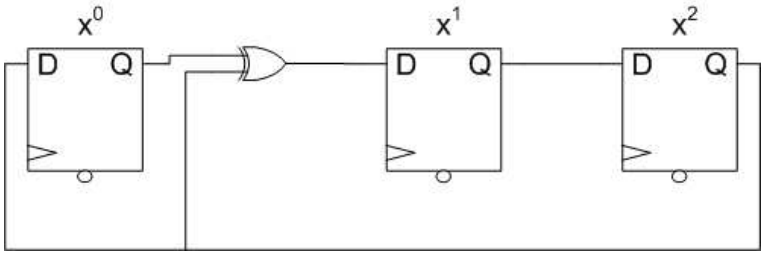

Figure 3.1: Circuit for the operation $xR(x) \pmod{f(x)}$

If the $R(x)$ is multiplied by $a_0 x$ then the result is

$$a_0 x R(x) = a_0 r_1 x^2 + a_0(r_0 + r_2)x + a_0 r_2 \pmod{f(x)} \tag{3.3}$$

where, for example, the term $a_0(r_0 + r_2)$ can be represented in gates as $a_0 \otimes (r_0 \oplus r_2)$, where $\otimes$ is an AND gate.

From the above, it can be seen that arithmetic primitive in $GF(2^m)$ can be implemented in hardware with D-type flip flops, XOR gates and AND gates. Primitives for other arithmetic operations are presented in Chapter 5.

3.3 Elliptic Curves

In the following discussion the theory of elliptic curve is described. Further information and proofs can be found in references [39, 101, 12]

3.3.1 Weierstrass Equation

Definition 3.5 (Weierstrass Equation). *An elliptic curve E over the field $GF(p^m)$ is given by the Weierstrass equation*

$$E(GF(p^m)) = y^2 + a_1xy + a_3y \equiv x^3 + a_2x^2 + a_4x + a_6 \tag{3.4}$$

where $a_i \in GF(p^m)$ and the discriminant $\Delta \neq 0$. Also $(x, y) \in GF(p^{mk}) \times GF(p^{mk})$ where m, k are integers. There is also an additional point on the curve known as the point at infinity, $\mathcal{O}$.

3.3.2 Supersingular and non-supersingular curves

Theorem 3.1 (Hasse). *The number of points on the curve is*

$$\#E(GF(p^m)) = p^m + 1 - t, \tag{3.5}$$

where t is the trace of Frobenius and is equal to

$$|t| \leq 2\sqrt{(p^m)}. \tag{3.6}$$

Definition 3.6 (supersingular). *If the trace of Frobenius, t, is divisible by the characteristic of $GF(p^m)$ then the elliptic curve is known as supersingular.*

This means that a supersingular curve has a group order of $p^m + 1$

Definition 3.7 (non-supersingular). *If the trace of Frobenius, t, is not divisible by the characteristic of $GF(p^m)$ then the elliptic curve is known as non-supersingular.*

The Weierstrass equation (3.4) can be simplified in the case where the characteristic of the underlying field is two. If $a_1 \neq 0$ then a non-supersingular elliptic curve E defined over $GF(2^m)$ is given in (3.7) such that $a, b \in GF(2^m)$. Also, $(x, y) \in GF(2^{mk}) \times GF(2^{mk})$ where m, k are integers.

$$y^2 + xy \equiv x^3 + ax^2 + b \tag{3.7}$$

A supersingular curve is defined when $a_1 = 0$ and is given by the equation

$$y^2 + cy \equiv x^3 + ax + b \tag{3.8}$$

where $a, b, c \in GF(2^m)$ and the discriminant $\Delta = c^4$.

Of particular interest to this work is the supersingular case and these are the curves that will be used for the rest of this thesis. For the curve considered in this thesis a, b and c are 1.

3.3.3 Elliptic curve arithmetic

The points of an elliptic curve form an Abelian group under addition. This addition operation is defined using a geometrical argument. If the curve is defined over the set of real numbers then it can be represented as in Figure 3.2. In this diagram the points P and R are added to give the result Q, which has coordinates (x_3, y_3). To calculate Q a line is drawn through points P and R, which have coordinates (x_1, y_1) and (x_2, y_2) respectively, then the equation for this line in entered into the equation of the curve and is solved for x. This value, x_3, is entered into the equation of the line to obtain y_3. Then the point Q' has

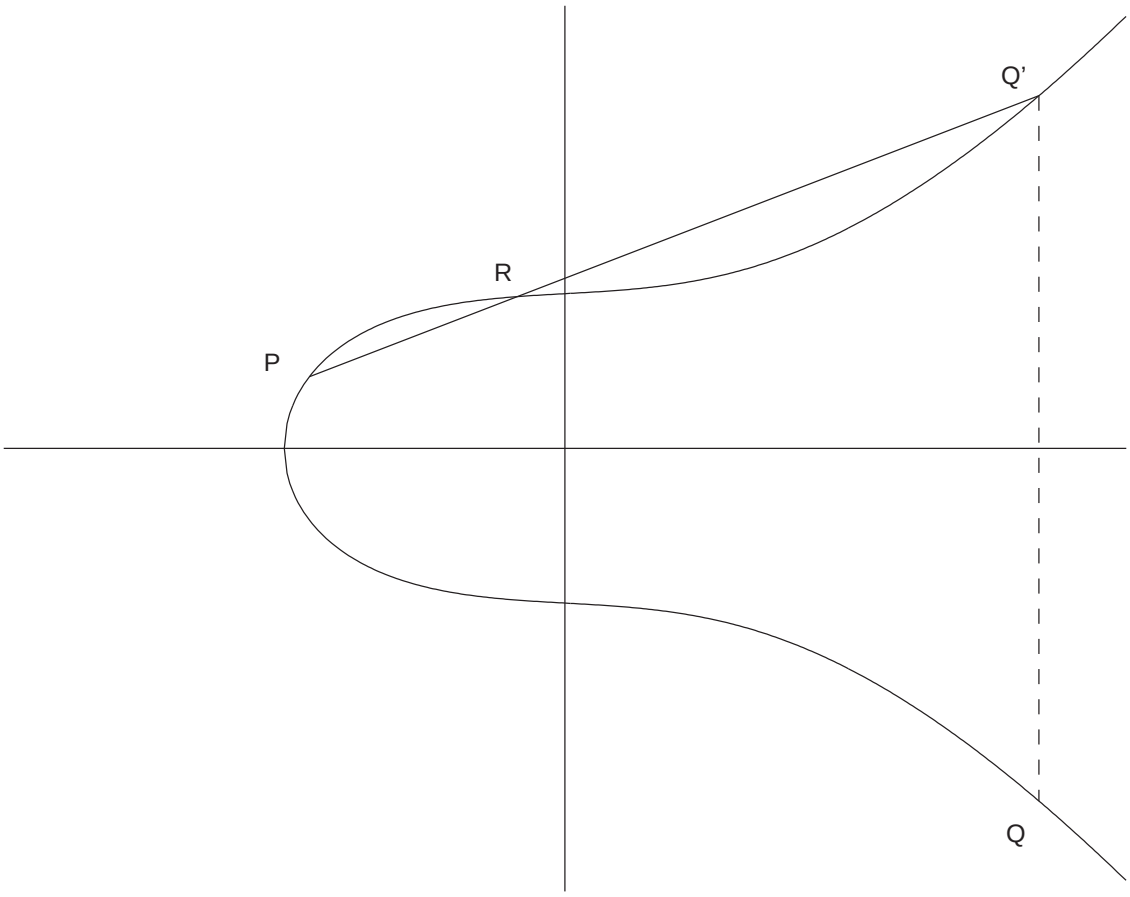

Figure 3.2: Point Addition

coordinates $(x_3, -y_3)$ and this point is reflected in the x axis to get the required result, Q.

When $P = R$ then the points have to be doubled (see Figure 3.3) and the technique is slightly different. A tangent is taken at a point P and the equation for this line is used as above to obtain the point Q.

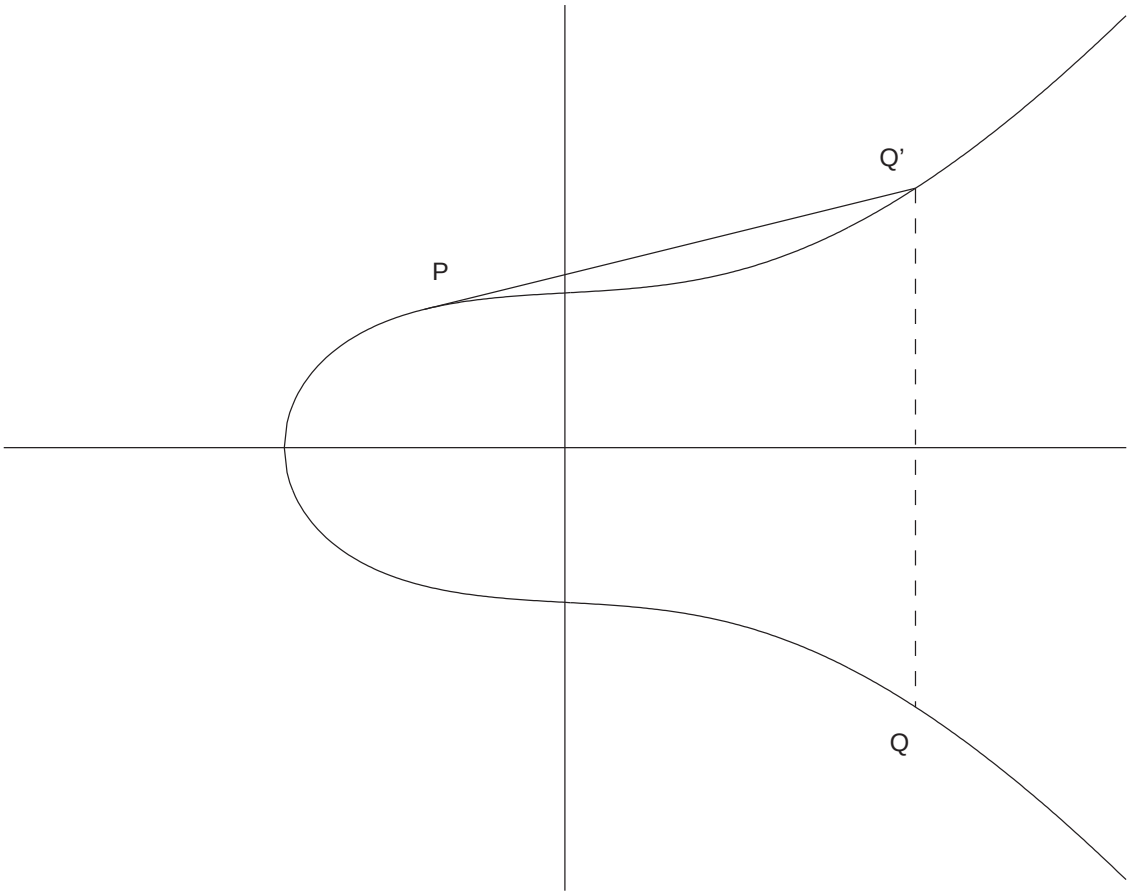

Figure 3.3: Point Doubling

In order for the points on the curve to form a group they have to have an identity element and an inverse. The identity element is defined as the point at infinity ($\mathcal{O}$) such that $P + \mathcal{O} = P$ and the inverse is the reflection of the point in the x axis such that $P + (-P) = \mathcal{O}$.

When working with a supersingular curve and when $P \neq R$, point addition is defined as

$$x_3 = \lambda^2 + x_1 + x_2, \tag{3.9}$$

$$y_3 = \lambda(x_1 + x_3) + y_1 + 1, \tag{3.10}$$

where $\lambda = \frac{y_1+y_2}{x_1+x_2}$ is the gradient of the line joining P and R.

When $P = R$ the equations for Q are defined as

$$x_3 = \lambda^2, \tag{3.11}$$

$$y_3 = \lambda(x_1 + x_3) + y_1 + 1, \tag{3.12}$$

where $\lambda = x_1^2 + 1$ is the gradient of the tangent at P obtained by implicit differentiation of the curve.

3.4 Divisor Theory

Divisors are introduced in this section and a more in depth discussion of the topic can be found in references [101, 63]

Definition 3.8 (Divisor). *A divisor D is a formal sum of points P on the curve $E(GF(2^{mk}))$*

$$D = \sum_{P \in E} n_p(P), \tag{3.13}$$

where n_p is an integer and equal to zero for all but finitely many P.

The degree of a divisor is given by Equation (3.14), and it is an integer value.

$$deg(D) = \sum n_p \tag{3.14}$$

The support of a divisor are the points on the curve such that n_p is not zero, and it is given by Equation (3.15).

$$supp(D) = \{P \in E \,|\, n_p \neq 0\} \tag{3.15}$$

The sum of a divisor is the value obtained when the points that make up the divisor

are added together. This is a point of the curve and it is given by Equation (3.16)

$$sum(D) = \sum n_p P \in E(GF(2^{mk})) \tag{3.16}$$

The set of divisors form a group where the group operation is addition and it is defined as

$$\sum_{P \in E} n_p(P) + \sum_{P \in E} m_p(P) = \sum_{P \in E} (n_p + m_p)(P). \tag{3.17}$$

The divisor of rational functions on the elliptic curve has n_p positive when the function evaluates to zero and n_p is negative when it evaluates to ∞ at a point P. The order of the point is calculated using the uniforming parameter for P.

Of particular interest to this thesis is the function l_1, which is the line from P to Q' in Figure 3.2 and the function l_2, which is the line Q' to Q in the same figure. The divisors of these functions are

$$div(l_1) = (P) + (R) + (Q') - 3\mathcal{O}. \tag{3.18}$$

$$div(l_2) = (Q) + (Q') - 2\mathcal{O}. \tag{3.19}$$

The degree of the these two divisors is equal to zero, and their sum equates to $\mathcal{O}$, which is always the case for a rational function on an elliptic curve. These divisors are known as principle and are denoted by D_l. D_l is a subgroup of D_0, which is the divisors of degree zero. The divisor of the product and quotient of the two functions are give below

$$div(l_1 l_2) = div(l_1) + div(l_2) = (P) + (R) + 2(Q') + (Q) + 5\mathcal{O}, \tag{3.20}$$

$$div\left(\frac{l_1}{l_2}\right) = div(l_1) - div(l_2) = (P) + (R) - (Q) - \mathcal{O}. \tag{3.21}$$

There is an equivalence relation between two divisors, $D_1, D_2 \in D_0$ such that $D_1 \equiv D_2$ if $D_1 = D_2 + div(f)$ where f is a function on the curve. Every $D \in D_0$ can be mapped to a point $Q = sum(D)$ such that $D \equiv (Q) - (\mathcal{O})$. With reference to Figure 3.2,

Equation (3.18) and Equation (3.19) if,

$$D_1 = (P) - (\mathcal{O}) + div(g_1), \tag{3.22}$$

$$D_2 = (R) - (\mathcal{O}) + div(g_2), \tag{3.23}$$

where g_1, g_2 are rational functions on the elliptic curve, then these two divisors add to give

$$\begin{aligned} D_1 + D_2 &= (P) + (R) - 2(\mathcal{O}) + div(g_1) + div(g_2) \\ &= (P) + (R) + (Q') - 3(\mathcal{O}) + (Q) - (\mathcal{O}) - (Q') - (Q) + 2(\mathcal{O}) + div(g_1) + div(g_2) \\ &= div(l_1) + (Q) - (\mathcal{O}) - div(l_2) + div(g_1) + div(g_2) \\ &= (Q) - (\mathcal{O}) + div\left(\frac{l_1}{l_2} g_1 g_2\right). \end{aligned} \tag{3.24}$$

A similar result can be found for point doubling.

3.5 Discrete Logarithm Problem

The Discrete Logarithm Problem (DLP) is the basis for the security of the algorithms used in this work and defined as;

Definition 3.9 (Discrete Logarithm Problem). *Given $f = g^x \in G$ and g is the generator element of the finite cyclic group G, the problem posed is to find the value of the integer x.*

In the context of this work the problem is posed over the fields $GF(2^{mk})$ and $E(GF(2^m))[l]$ where it is known as the Elliptic Curve Discrete Logarithm Problem (ECDLP). The ECDLP is given two points on the elliptic curve xP and P, where P generates a finite cyclic group, what is the integer x.

3.6 The Tate Pairing

If $l \mid \#E(GF(2^m)$ where l is a large prime l, and k is the smallest integer such that $l \mid (2^{mk} - 1)$ then the Tate pairing is defined as [101, 11];

Definition 3.10 (The Tate Pairing). *The Tate pairing, e_l, is the mapping*

$$\begin{aligned} e_l &= E(GF(2^{mk}))[l] \times E(GF(2^{mk}))/lE(GF(2^{mk})) \rightarrow (GF(2^{mk}))^*/(GF(2^{mk}))^{*l} \\ &= \mathbb{G}_1 \times \mathbb{G}_2 \rightarrow \mathbb{G}_T \end{aligned} \tag{3.25}$$

where the l torsion points, $\mathbb{G}_1$, are

$$E(GF(2^{mk}))[l] = \{P \in E(GF(2^{mk})) \mid lP = \mathcal{O}\}. \tag{3.26}$$

And two points $P, Q \in E(GF(2^{mk}))$ are members of the same equivalence class, $\mathbb{G}_2$, if

$$P \equiv Q \pmod{E(GF(2^{mk}))/lE(GF(2^{mk}))} \tag{3.27}$$

i.e. $P = Q + lR$ where $R \in E(GF(2^{mk}))$. Similarly $a, b \in (GF(2^{mk}))^*$ are members of the same equivalence class, $\mathbb{G}_T$, such that

$$a \equiv b \pmod{(GF(2^{mk}))^*/(GF(2^{mk}))^{*l}} \tag{3.28}$$

which can be also be stated as $a = bc^l$ for $c \in (GF(2^{mk}))^*$.

Its most desirable property in the context of cryptography is bilinearity

$$e_l(aP, bQ) \equiv e_l(aP, Q)^b \equiv e_l(P, bQ)^a \equiv e_l(P, Q)^{ab} \tag{3.29}$$

where a, b are integers. The exponent $\frac{2^{mk}-1}{l}$ of the output of the pairing to provides a unique value rather than a member of an equivalence class. The integer k is known as the security multiplier and is four for the particular curve considered in this thesis.

The Tate pairing essentially takes two points on an elliptic curve and maps them to

a element of a multiplicative group of a large finite extension field. The choice of the elliptic curve group over which the ECDLP is posed must be such that it requires at least 2^{80} operations to solve. Therefore l has to be at least of the order of $\approx 2^{160}$ (see Section 4.4.3). Also, the finite field to which the Tate pairing maps must be sufficiently large to make the DLP intractable i.e. that it has a running time of 2^{80}. For a binary field, as used in this thesis, it has to be of the order of 2^{1024} . From equation (4.4) this means that running time for solving the DLP is 2^{84}.As $k = 4$ for the curve used in this thesis, this means that m must be at least 250. It was initially decided to use a NIST approved curve and therefore m has been selected as 283 [71].

3.7 Algorithms for calculating the Tate pairing

Let $D_P \equiv (P) - (\mathcal{O})$ where $P \in E(GF(2^{mk})[l]$. A function on the curve f_P with divisor $div(f_P)$ is defined by

$$div(f_P) = lD_P = l(P) - l(\mathcal{O}) \tag{3.30}$$

as $lP = \mathcal{O}$ and the degree of $div(f_P)$ is zero. If another divisor $D_Q \in D^0$ i.e. $D_Q \equiv \sum_{R \in E} m_r(R) \equiv (Q) - (\mathcal{O})$ where $Q \in E(GF(2^{mk}))/lE(GF(2^{mk}))$, and D_P and D_Q have disjoint support, then the Tate pairing can be defined as

$$e_l(P, Q) = f_P(D_Q) = \prod f_P(R)^{m_r}. \tag{3.31}$$

Their divisors have to have disjoint support or else Equation (3.31) would equate to zero.

Given $D_P = (P) - (\mathcal{O}) + div(1)$ and Equation (3.24) then f_P can be calculated by

$$\begin{aligned}
2D_P &= (P+P) - (\mathcal{O}) + div(\frac{l_1}{l_2}1.1) = (2P) - (\mathcal{O}) + div(\frac{f_1}{f_2}) \\
4D_P &= (2P+2P) - (\mathcal{O}) + div(\frac{l_1}{l_2}\frac{f_1^2}{f_2^2}) = (4P) - (\mathcal{O}) + div(\frac{f_1}{f_2}) \\
5D_P &= (4P+P) - (\mathcal{O}) + div(\frac{l_1}{l_2}\frac{f_1}{f_2}.1) = (5P) - (\mathcal{O}) + div(\frac{f_1}{f_2}) \\
&\vdots \\
lD_P &= ((l-1)P+P) - (\mathcal{O}) + div(\frac{l_1}{l_2}\frac{f_1}{f_2}.1) = div(\frac{f_1}{f_2})
\end{aligned} \tag{3.32}$$

3.7.1 Miller's Algorithm

Miller's algorithm [64] (see Algorithm 1) uses the above technique, given in Equation (3.32), to calculate the Tate pairing. By Equation (3.31) the output that is required is $f_P((Q) - (\mathcal{O}))$. Instead of waiting until the f_P is found and then inserting D_Q into the function, the intermediate functions are calculated at D_Q. In order to prevent an intermediate function evaluating to zero, i.e. D_Q is in the support of the function, a random point on the curve, $S \in E(GF(2^{mk})$, is chosen so as to generate $Q' = Q + S$. Therefore $D_Q \equiv (Q') - (S) \equiv (Q) - (\mathcal{O})$ and it is at these values the intermediate functions are calculated. If T has coordinates (t_x, t_y) and $U = 2T$ has coordinates (u_x, u_y) then

Algorithm 1 Miller's Algorithm [64]

$n = \lfloor \log_2 l \rfloor - 1$
$Q' = Q + S$ where random $S \in E(GF(2^{mk})$
$f = 1, T = P$
for n downto 1 **do**
  $T = 2T$
  $f = f^2 \frac{(l_{2T}(Q')l_{2T-2T}(S))}{(l_{2T}(S)l_{2T-2T}(Q'))}$
  if n^{th} bit of $l = 1$ **then**
    $T = T + P$
    $f = f \frac{(l_{T+P}(Q')l_{(T+P)-(T+P)}(S))}{(l_{T+P}(S)l_{(T+P)-(T+P)}(Q'))}$
  end if
end for
return $f_P(D_Q) = f$

$$\begin{aligned} u_x &= \lambda^2 \\ u_y &= \lambda(t_x + u_x) + t_y + 1 \end{aligned} \tag{3.33}$$

where $\lambda = t_x^2 + 1$ is the gradient of the tangent at T. Also if Q' and S have coordinates (q'_x, q'_y) and (s_x, s_y) respectively, then

$$\begin{aligned} l_{2T}(Q') &= q'_y + \lambda(t_x + q'_x) + t_y \\ l_{2T}(S) &= s_y + \lambda(t_x + s_x) + t_y \end{aligned} \tag{3.34}$$

and

$$\begin{aligned} l_{2T-2T}(Q') &= q'_x + u_x \\ l_{2T-2T}(S) &= s_x + u_x \end{aligned} \tag{3.35}$$

If $U = T + P$ and P has coordinates (p_x, p_y) then

$$\begin{aligned} u_x &= \lambda^2 + t_x + p_x \\ u_y &= \lambda(t_x + u_x) + t_y + 1 \end{aligned} \tag{3.36}$$

where $\lambda = \frac{t_y + p_y}{t_x + p_x}$ is the gradient of the line joining T and P. And the line functions are evaluated at

$$\begin{aligned} l_{T+P}(Q') &= q'_y + \lambda(t_x + q'_x) + t_y \\ l_{T+P}(S) &= s_y + \lambda(t_x + s_x) + t_y \end{aligned} \tag{3.37}$$

and

$$\begin{aligned} l_{(T+P)-(T+P)}(Q') &= q'_x + u_x \\ l_{(T+P)-(T+P)}(S) &= s_x + u_x \end{aligned} \tag{3.38}$$

If the n^{th} bit of $l = 0$, then this algorithm requires six multiplications and one inversions in $GF(2^{mk})$ in the loop, and these are both costly operations – see Chapter 5.

3.7.2 BKLS/GHS Algorithm

The BKLS/GHS algorithm [33, 8] improves upon Miller's algorithm. If $P \in E(GF(2^m)[l]$ and $Q \in E(GF(2^m))[l]$ the modified Tate pairing is $e_l(P, \phi(Q))^{\frac{2^{mk}-1}{l}}$ where ϕ is a distortion map and the final value is raised to the value $\frac{2^{mk}-1}{l}$ to provide a unique value of the l^{th} roots of unity , μ_l, rather than a member of an equivalence class. This pairing is symmetric with $\mathbb{G}_1 = \mathbb{G}_2 = \mathbb{G}$.

$$e_l = \mathbb{G} \times \mathbb{G} \rightarrow \mu_l \tag{3.39}$$

As $P \in E(GF(2^m)[l]$ this means that all the point doubling and adding takes place in the field $GF(2^m)$ rather than $GF(2^{mk})$, which reduces the amount of computation required. Also, the intermediate functions are in this subfield, which means that multiplication by λ requires less resources – see Equation (3.34) and (3.37).

It is demonstrated in [8] that the final exponentiation ,$\frac{2^{mk}-1}{l}$, has a factor $2^m - 1$ and that

$$e_l(P, Q) = f_P(D_Q) = f_p((Q) - (\mathcal{O}))^{\frac{2^{mk}-1}{l}} = f_p((Q))^{\frac{2^{mk}-1}{l}} \tag{3.40}$$

This is because $f_p((\mathcal{O})) \in GF(2^m)$ and therefore due to Fermat's little theorem is equal to the identity element (see Equation (3.41))

$$f_p((\mathcal{O}))^{2^m-1} \equiv 1 \pmod{f(x)} \tag{3.41}$$

where $f(x)$ is used to generate the field $GF(2^m)$.

The distortion map ϕ takes a point on $E(GF(2^m))$ and maps it to a point on $E(GF(2^{mk}))$

$$\phi(q_x, q_y) = (q_x + s^2, q_y + sq_x + t) \tag{3.42}$$

where $s^2 + s + 1 = 0$ and $t^2 + t + s = 0$ for the curve considered in this thesis. It is shown in [8] that the final exponentiation also means that the functions $l_{2T-2T}(\phi(Q))^{\frac{2^{mk}-1}{l}}$

and $l_{(T+P)-(T+P)}(\phi(Q))^{\frac{2^{mk}-1}{l}}$ evaluate to unity.

$$\begin{aligned} l_{2T-2T}(\phi(Q))^{\frac{2^{mk}-1}{l}} &= 1 \\ l_{(T+P)-(T+P)}(\phi(Q))^{\frac{2^{mk}-1}{l}} &= 1 \end{aligned} \tag{3.43}$$

The curve that is being used is

$$E(GF(2^{283})) = y^2 + y \equiv x^3 + x + 1 \tag{3.44}$$

and $l = \#E(GF(2^m) = 2^{283} + 2^{142} + 1$, which has a small Hamming weight and this reduces the number of times that the "add" branch of the loop is entered.

Algorithm 2 BKLS/GHS Algorithm [33, 8]

$n = \lfloor \log_2 l \rfloor - 1$
$Q' = \phi(Q)$
$f = 1, T = P$
for n downto 1 **do**
  $T = 2T$
  $f = f^2(l_{2T}(Q'))$
  if n^{th} bit of $l = 1$ **then**
    $T = T + P$
    $f = f(l_{T+P}(Q'))$
  end if
end for
return $f^{\frac{2^{mk}-1}{l}}$

Neglecting the "add" branch of the loop, as it is only entered three times, then this algorithm requires only one multiplications between elements in $GF(2^{mk})$, which is an obvious improvement on Algorithm 1. The calculation of $2T$ only requires one multiplication in $GF(2^m)$ (see Equation (3.33)) and $l_{2T}(Q')$ requires four multiplications as $\lambda \in GF(2^m)$ and $(t_x + q'_x) \in GF(2^{mk})$.

$$l_{2T}(Q') = q'_y + \lambda(t_x + q'_x) + t_y \tag{3.45}$$

The operations $f = f^2(l_{2T}(Q'))$, where $f \in GF(2^{mk})$, requires four squarings in $GF(2^m)$, followed by a multiplication in $GF(2^{mk})$ which can be accomplished with nine multiplications in $GF(2^m)$ (see Section 5.5.3). Therefore, every time the "double" branch

of the loop is entered fourteen $GF(2^m)$ multiplications are computed.

3.7.3 η Algorithm

Based upon the work of Duursma and Lee [27], a closed form of the Tate pairing calculation, which is known as the η algorithm, has been obtained for characteristic two [59, 6]. The Tate pairing is given by

$$f_p(\phi(Q))^{\frac{2^{mk}-1}{l}} = g_p(\phi(Q))^{2^{2m}-1} \tag{3.46}$$

where

$$g_p = \prod_{i=1}^{2m} l_{2^iP}^{2^{2m-i}} \tag{3.47}$$

and l_{2^iP} is the equation of the tangent to the curve at the point 2^iP. Through application of the distortion map and a lot of algebraic manipulation Equation (3.48) is arrived at and this is rewritten in the form of Algorithm 3.

$$g_p(\phi(Q)) = \prod_{i=1}^{m} p_x^{2^i} q_x^{2^{(-i+1)}} + p_y^{2^i} + q_y^{2^{(-i+1)}} + s^2(p_x^{2^i} + q_x^{2^{(-i+1)}}) + t^2 + b \tag{3.48}$$

This algorithm requires seven multiplications in the field $GF(2^m)$, a big improvement

Algorithm 3 η Algorithm [59, 6]

$P = (p_x, p_y), Q = (q_x, q_y)$
$C(x) = c_3x^3 + c_2x^2 + c_1x + c_0 = 1$
for $i = 1$ to m **do**
  $p_x = p_x^2$
  $p_y = p_y^2$
  $z = p_x + q_x$
  $w = z + p_xq_x + p_y + q_y + 1$
  $c_0 = c_0w + (c_2 + c_3)(z + 1) + c_3$
  $c_1 = c_0w + (c_1 + c_2 + c_3)w + (c_0 + c_2 + c_3)(w + z + 1) + c_3(z + 1) + c_0 + c_3$
  $c_2 = c_0w + (c_1 + c_2 + c_3)w + (c_0 + c_2 + c_3)(w + z + 1) + (c_1 + c_2)(w + z + 1) + c_1$
  $c_3 = (c_1 + c_2 + c_3)w + (c_1 + c_2)(w + z + 1) + c_2$
  $q_x = q_x^{2^{m-1}}$
  $q_y = q_y^{2^{m-1}}$
end for
return $f_p(\phi(Q))^{\frac{2^{mk}-1}{l}} = C(x)^{2^{2m}-1}$

on the previous algorithms. It has a regular structure that maps well to hardware, and so it will be the Tate pairing algorithm that will be implemented in this thesis. Another approach would have been to use the η_T algorithm to implement the Tate pairing [6]. This algorithm has half the loop of the η but a more complicated final exponentiation which is given in equation (3.49).

$$C(x) = C(x)^{(2^{2m}-1)(2^{2m}+2^{\frac{m+1}{2}}+1)(2^{\frac{m+1}{2}}-1)} \tag{3.49}$$

This final exponentiation require three Frobenius operations, $m+1$ squarings, four multiplications and two inversions all taking place in the finite extension field. When selecting an algorithm to implement it was judged that this would lead to a design with more complicated datapath and control logic, and also a larger design that could be achieved by using the η algorithm. This would seem to be confirmed by the recent results of Shu [93] (see Table 6.4). As future work it should be implemented to compare with the η approach.

3.8 Bilinear Diffie Hellman Problem

The identity based cryptosystems discussed later in this thesis are based on the hardness of the Bilinear Diffie Hellman Problem (BDHP).

Definition 3.11 (Bilinear Diffie Hellman Problem). *Given* $P, aP, bP, cP \in E(GF(2^m))$ *it is computationally infeasible to calculate* $e_l(P,P)^{abc} \in GF(2^{mk})$

This implies that the Diffie Hellman Problem (DHP) is also a hard problem in $E(GF(2^m))$ and $GF(2^{mk})$. The DHP in $E(GF(2^m))$ is that given P, aP and bP it is hard to calculate abP. If this was not the case then abP and cP could be input to the Tate pairing to calculate $e_l(P,P)^{abc} = e_l(abP, cP)$, and so the BDHP is solved.

The DHP in $GF(2^{mk})$ is given three elements α, α^x and $\alpha^y \in GF(2^{mk})$ it is computationally infeasible to calculate $\alpha^{xy} \in GF(2^{mk})$ if the field is large enough. Also, if the DHP is easy in $GF(2^{mk})$ then the BDHP can be solved. For given $e_l(aP, bP) = e_l(P,P)^{ab}$ and $e_l(P, cP) = e_l(P,P)^c$ then $e_l(P,P)^{abc}$ can be calculated.

3.9 Summary

The cryptographic protocol outlined in Chapter 4 makes use of the Tate pairing, which is a bilinear map defined over an elliptic curve. It takes two points on a elliptic curve, that have coordinates in $GF(2^m)$, and maps them to a element of the l roots of unity in $GF(2^{mk})$. The underlying field $GF(2^m)$ on which the elliptic curve is defined should be large enough so that the ECDLP is computationally infeasible, this means that it should be of the order of at least 2^{160} – see Chapter 4. The DHP is posed in the field to which the Tate pairing maps these points, and in order for the problem to be secure the size of the field has to be at least 2^{1000}. As the security multiplier, k, is four for the curve considered in this thesis, then m has to be at least 250. In fact, the curve that is being used is $E(GF(2^{283}))$ – see Equation (3.44)

The Tate pairing was originally calculated using Miller's algorithm [64]. Improvements have recently been proposed to Miller's algorithm to reduce its cost in terms of its execution time [33, 8, 59, 6]. Of particular interest in this thesis is the η algorithm proposed in [59, 6] which is the most amenable to a low energy hardware implementation, as it is has the least number of multiplications in the subfield, is defined over a field of characteristic two and has a regular structure which maps well to hardware (see Algorithm 3).

CHAPTER 4

Security Considerations in a WSN

4.1 Introduction

There is a clear need for security in a wireless sensor network. The main requirements, known as confidentiality and authentication respectively, are that the data exchanged in the network should not be read by an unauthorised third party and also that this third party cannot join the network.

Confidentiality can be assured by data encryption, either with an asymmetric or symmetric cipher. In practice, an asymmetric system is considered too costly in terms of its computational complexity to perform any task other than the distribution of symmetric keys, or sending one off messages of the order of a symmetric key . In order to perform network access control the wireless sensor nodes should be able to generate digital signatures.

4.2 Symmetric Systems

In symmetric key cryptosystems security is dependent on keeping the key that is shared between the communicating wireless sensor nodes a secret. This means that an adversary should not be able to obtain the key, and that if it does then the security of the system is compromised.

Figure 4.1 shows Alice encrypting a message, m, with the encryption algorithm, E, and the secret key, k. The adversary, Eve, can only see the ciphertext, c. Bob uses the publicly available decryption algorithm, D, together with the secret key to extract the message.

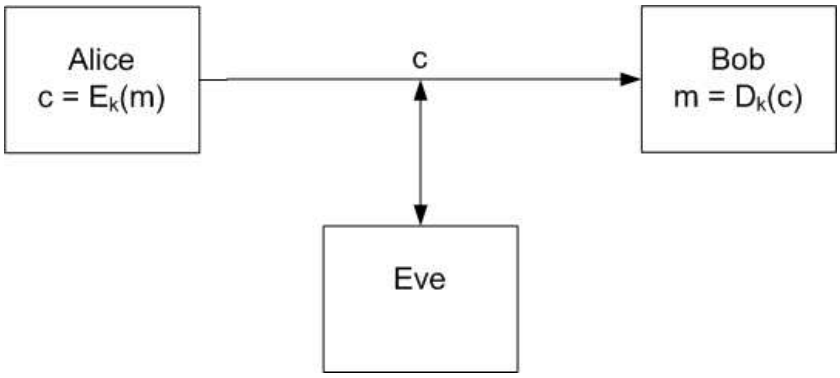

Figure 4.1: Symmetric Cryptosystem

The feasibility of a symmetric scheme for wireless sensor networks has been proved [51]. In this scheme a software approach has been shown to only add a 10% overhead of energy, though it should be noted that the key distribution problem is not addressed in this scheme.

4.3 Key Distribution

Of the many challenges facing real deployments of WSNs, the distribution of symmetric keys in the network is one of the most difficult to address. As there is no in-situ infrastructure for the wireless sensor nodes to interact with in order to obtain keys on the fly, like in a traditional Public Key Infrastructure (PKI) approach, novel techniques have to be employed. This section discusses different approaches to the key distribution problem.

4.3.1 System-Wide Key

The simplest approach to deploy a symmetric system would be that all the wireless sensor nodes share the same key. As the wireless sensor nodes could be placed in a region where an adversary can capture them, it is likely that it could extract the secret key, and therefore

would be able to monitor all communication in the network. For this reason, this method of ensuring privacy is not appropriate in a hostile environment.

4.3.2 Pair-Wise Keys

Another method would be for all the wireless sensor nodes to set up pair-wise keys between them before deployment. If there are n wireless sensor nodes in the network then each wireless sensor node would have to store $n - 1$ keys in its persistent memory. In a resource constrained device this would be a problem as storing the keys would use too much memory. The other main drawback to using this scheme is that it does not scale. If, after deploying the bulk of the wireless sensor nodes, it is required to add extra wireless sensor nodes then this is not possible unless the extra wireless sensor nodes' keys are already programmed in the deployed network. Upon capture of a wireless sensor node, however, only its $n - 1$ links will be compromised, which is a slight improvement on the system that uses only one symmetric key.

4.3.3 Probabilistic Key Sharing

Yet another symmetric technique is probabilistic key sharing [29]. In this approach a large pool of keys is generated from which a smaller ring of keys is randomly selected and preloaded before deployment into each wireless sensor node. Each wireless sensor node thus has a separate ring of keys in which there may be a shared key. During the shared key discovery phase of the algorithm, the wireless sensor nodes ascertain whether or not there is secure path between them. It has been shown [29] that in order to create a network of 10,000 wireless sensor nodes the pool of keys has to be 100,000 and the key ring only has to be 250. This system is scalable as when a new wireless sensor node is added to network it only has to be preloaded with a random selection of 250 keys from the key pool. However, this scheme is not secure against capture by an adversary. If one wireless sensor node is captured then there is a probability of $\frac{250}{100000}$ that any of the links in the network can be deciphered, but this increases to $\frac{1}{10}$ after the capture of 40 wireless sensor nodes.

4.3.4 Public Key Systems

Another approach to key distribution is to employ an asymmetric or public key system. In these schemes there is a private/public key pair and it is considered computationally infeasible to calculate the private key from the public one. The wireless sensor nodes can be deployed with an embedded private/public key pair. They then broadcast the public key to their neighbours who can then use this public key to encrypt a message to them. This scheme has the added advantage that private key can be used to generate digital signatures. Asymmetric systems are secure against individual wireless sensor node capture and they are also scalable.

4.4 Asymmetric Approaches

From the discussion of Section 4.3 an asymmetric or public key cryptosystem would seem to be the best approach for distributing symmetric keys in a WSN. The security of these systems is ensured on the basis of a hard number theoretic problem, some of which are outlined below.

4.4.1 RSA System

RSA's [80] security is based on the difficulty of factoring a large integer. When two 512 bit numbers such as p and q are multiplied together to create a 1024 bit number n, there is no known polynomial time algorithm to factor this number.

$$n = p.q \tag{4.1}$$

4.4.2 El Gamal System

El Gamal's security [28] is based on solving the DLP in a finite field. Let $\alpha, \beta \in \mathbb{Z}_p$, then β is calculated as

$$\beta \equiv \alpha^y \pmod{p} \tag{4.2}$$

where y is an integer and α is a generator element of $\mathbb{Z}_p$ and p is a large prime number. The fastest algorithm for solving the DLP given α and β is based on the index calculus algorithm and is known as the function field sieve [1]. This works by selecting a number of small prime numbers from which their discrete logarithmic can be found with regard to α. The DLP can now be solved as the element α^y can be generated by multiplying together elements in the factors base, and their exponents can be added to give the required result. The running time for this algorithm is

$$\mathrm{O}(exp(1.923n^{1/3}ln^{2/3}n)) \tag{4.3}$$

In the case where $\alpha, \beta \in GF(2^m)$ and

$$\beta \equiv \alpha^y \pmod{p(x)} \tag{4.4}$$

where y is an integer and α is a generator element of $GF(2^m)$. Then y can be calculated using Coppersmith's algorithm [23]. The running time for this algorithm is

$$\mathrm{O}(exp(1.587n^{1/3}ln^{2/3}n)) \tag{4.5}$$

4.4.3 Elliptic Curve Cryptography

Elliptic Curve Cryptosystems (ECC) [39] are based on the ECDLP, which is defined as follows; given two points P and Q such that $Q = sP$, then what is the value of the integer s. The Index Calculus approach cannot be taken as one cannot factorise elliptic curve points, the elements in the group, and therefore the best algorithm for solving the ECDLP is Pollard's ρ method [75]. This algorithm employs a random function to generate a point in the group of order n that can be written as $R_1 = x_1P + y_1Q$, which is then input to the algorithm to generate another point $R_2 = x_2P + y_2Q$. All of these points are stored and when two points are the same then the ECDLP can be solved. If $R_1 = R_2$ and $Q = sP$

then;

$$\begin{aligned} x_1P + y_1Q &= x_2P + y_2Q \\ x_1P + y_1sP &= x_2P + y_2sP \\ (y_1 - y_2)sP &= (x_1 - x_2)P \\ \therefore s &= (x_1 - x_2)(y_1 - y_2)^{-1} \pmod{n} \end{aligned} \tag{4.6}$$

This algorithm has a running time of $\mathrm{O}(\sqrt{n})$.

Therefore, the operands used in ECDLP can be smaller than the ones used in the DLP, which is set over a multiplicative group in a finite field. If it is considered infeasible for an adversary to perform 2^{80} operations then n must be at least 2^{160} in order that the ECDLP provides sufficient security. The different key sizes required for the different approaches are presented in Table 4.4.3. From this, it can be seen that ECC only requires a key size of 163 bits to have the same level of security as 1024 bit El Gamal / RSA. ECC would seem to be the appropriate choice for WSNs. This smaller key size translates directly into an energy saving for the device, as fewer bits are required to be transmitted by the radio. The smaller operand size will also lead to a reduction in energy dissipated by the digital circuitry. However, in order to deploy any of these approaches, a PKI is required, see Section 4.5, and this poses significant difficulties in the context of WSNs.

ECC key	El Gamal	Security strength
163	1024	80
283	3072	128
409	7680	192
571	15360	256

Table 4.1: Equivalent key sizes in bits for ECC and El Gamal [71]

4.5 Public Key Infrastructure

In order to deploy a traditional asymmetric cryptosystem there has to be a PKI and Certificate Authority (CA) in place [58] that is responsible for providing digital certificates.

These certificates, which bind a wireless sensor node's identity to its public key, are then used by the wireless sensor nodes to prove that they have a valid public key. In Figure 4.2 the CA issues the keys kbob_public and kalice_public to alice and bob, respectively. These keys are incorporated in a digital certificate that can be verified with CA's public key and are then used with the encryption, E, and decryption algorithms, D, to securely send a message, m.

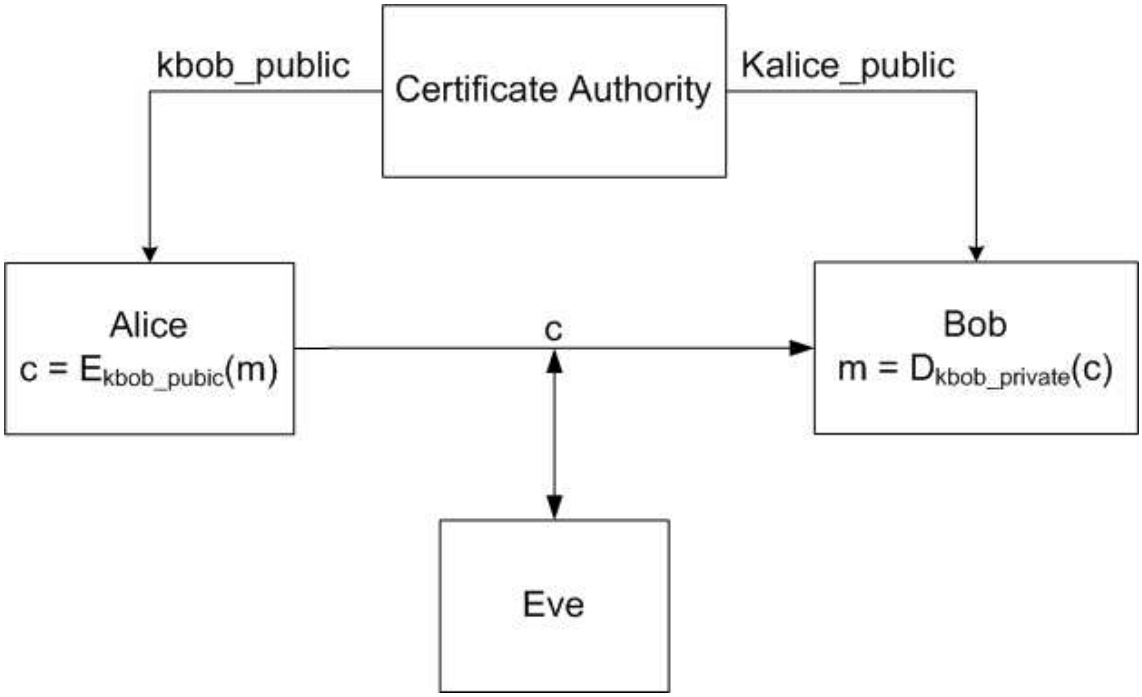

Figure 4.2: Public Key Infrastructure

In the context of WSNs, it is envisaged that each wireless sensor node will have stored the CA's public key, its private key and a digital certificate issued by the CA for its public key. The CA in this system would be the base station that originally programmed the wireless sensor nodes and extracts data from the network - see Figure 4.3.

The steps required for secure communication are given in Algorithm 4. The message in this case is a key for a symmetric cryptosystem. Steps one and five require that the wireless sensor nodes communicate with each other. As the radio is likely to be the main consumer of energy in the wireless sensor node, it is important to minimize the number of bits transmitted.

Unauthorised message recovery and malicious packet injection into the network is prevented using this algorithm. Messages will only be encrypted and sent to the wireless

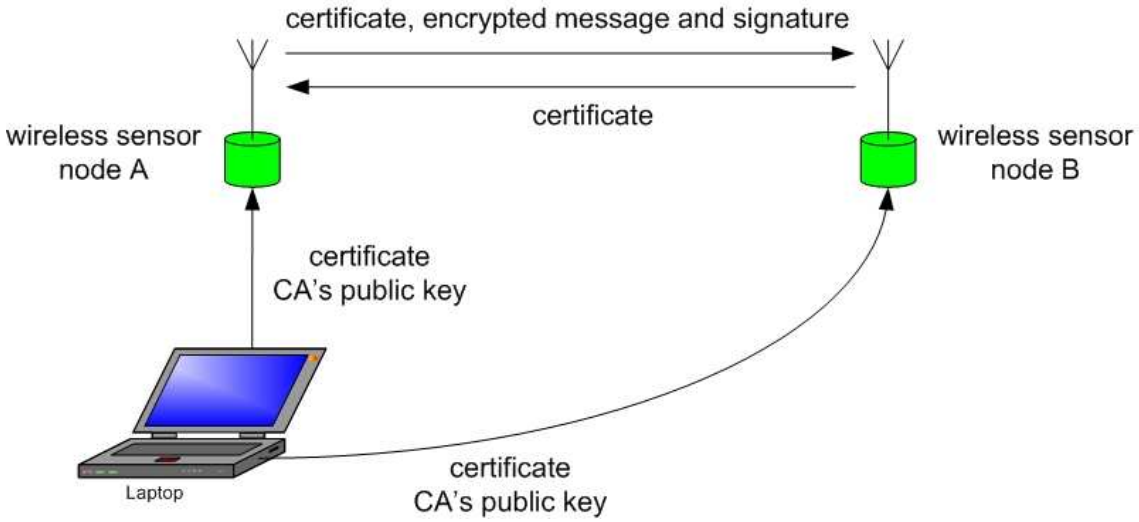

Figure 4.3: Deploying PKI in a WSN

Algorithm 4 PKI Data Transfer

1: Alice sends Bob her digital certificate.
2: Bob verifies the certificate using the CA's public key.
3: Bob signs his message using private key.
4: Bob encrypts the message using Alice's public key.
5: Bob sends the encrypted message, digital signature and his digital certificate to Alice.
6: Alice verifies Bob's digital certificate using the CA's public key.
7: Alice decrypts the message using her private key.
8: Alice verifies the digital signature using Bob's public key.

sensor nodes from which valid digital certificates have been received. Similarly, messages will only be accepted if they have a valid digital signature and this can only be verified if a valid digital certificate has been received. Since only wireless sensor nodes that are programmed by the base station will have valid private key / digital certificate pairs, access control is achieved.

Also, this scheme is easily scalable as new wireless sensor nodes just need to be programmed with a valid private key and digital certificate by the base station before deployment to communicate with the other wireless sensor nodes in the network.

A traditional ECC technique such as the Elliptic Curve Integrated Encryption Scheme (ECIES) [39] has a ciphertext (R, C, t) where R is a point in the curve $E(GF(2^{163}))$, C is the output of the Advanced Encryption Standard (AES) algorithm and t is a message authentication code. In this case R is 164 bits, if point compression is used i.e. the x-coordinate is transmitted and also one bit to indicate which of the two values of the y-coordinate is valid. C is 128 bits and t is 160 bits. The total number of bits in the ciphertext is 452. Elliptic Curve Digital Signature Algorithm (ECDSA) [39] would be an appropriate choice for the signing algorithm. The numbers of bits transmitted in the signature is 320.

From Algorithm 4 it can be seen that two digital certificates are sent. The certificates are a point on the curve and the wireless sensor nodes identity signed using ECDSA and therefore are 494 bits long (164 + 10 + 320), if the identify of the wireless sensor node is 10 bits. The total number of bits transmitted in a communication round is 1670.

4.6 Identity-Based Cryptography

The concept of IBE was proposed by Shamir in 1984 [89], but it was not until 2001 that a workable scheme was presented. This is the Boneh-Franklin scheme [13] and the hard number problem on which this system is based is the BDHP. The protocol was originally designed using the Weil pairing but the Tate pairing can be used instead.

4.6.1 Identity-Based Encryption

In IBE there is a requirement for a Key Generation Centre (KGC). Its role is to generate the domain parameters and private keys, which are distributed to Alice and Bob (see Figure 4.4). The public keys are calculated from the domain parameters and the publicly available identities. The Boneh-Franklin IBE scheme's [13] security is based on the

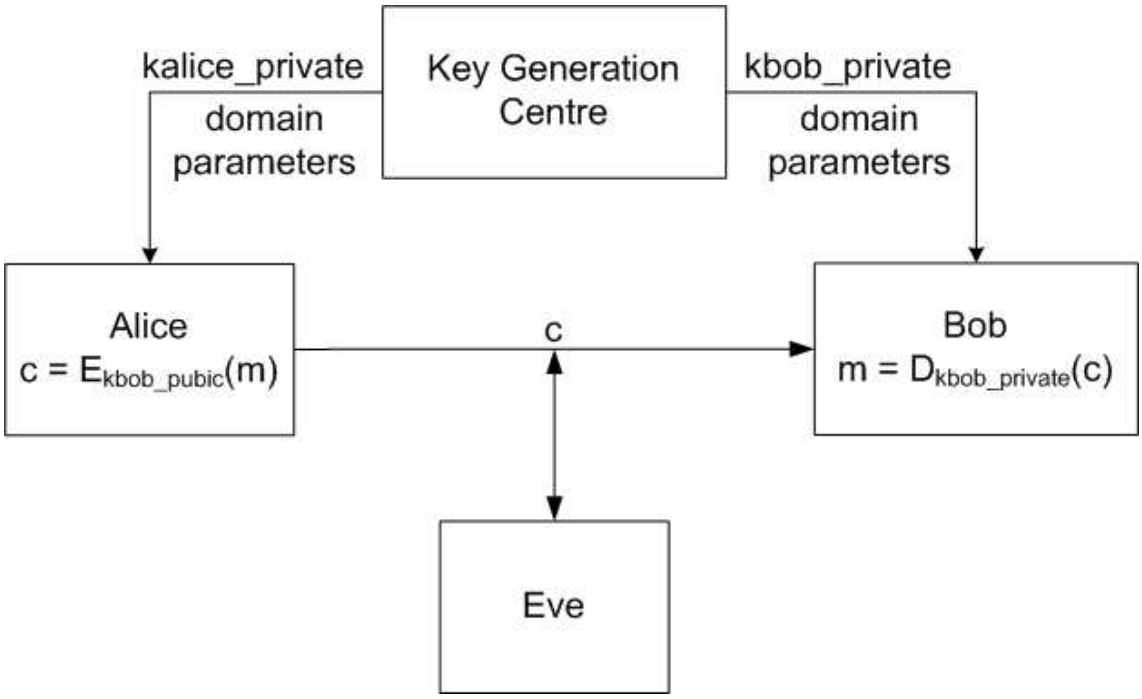

Figure 4.4: Identity-Based Encryption

BDHP and it requires four algorithms; setup, extract, encrypt and decrypt. "Setup" generates the master key, and the domain parameters. "Extract" generates the private key. "Encrypt" and "Decrypt" are the encryption and decryption algorithm respectively.

Setup The domain parameters are generated by this algorithm. $l \in \mathbb{Z}$, of the order of 2^{160}, is chosen. An integer $s \in \mathbb{Z}_l^*$ is multiplied by $P \in \mathbb{G}_1$ to generate sP. Four hash functions are chosen;

- $h_1 : \{0,1\}^* \rightarrow \mathbb{G}_1$
- $h_2 : GF(2^{283\times4}) \rightarrow \{0,1\}^n$
- $h_3 : \{0,1\}^n \times \{0,1\}^n \rightarrow \mathbb{Z}_l^*$
- $h_4 : \{0,1\}^n \rightarrow \{0,1\}^n$

where n is the length of the message to be encrypted. h_1 is a probabilistic algorithm to calculate a point on the curve. It takes as its input the identity (i.d.) of the recipient and maps this value to a element of the field $GF(2^{283})$, which is taken as x and substituted into Equation (3.44). If the equation holds then it is a point on the curve, otherwise x is incremented until a point is found which is denoted by Q. This point is then multiplied by $\#E/l$ to get an l-torsion point. All of this information is public and is placed on the wireless sensor node except the master key, s.

Extract The hash function, $h_1(i.d.) = Q$, is run to map an identity to a point on the curve. When this value is multiplied by s and the result, sQ, is the private key.

Encrypt In order for Bob to encrypt a message, $m \in \{0,1\}^n$, which he sends to Alice, he performs the following steps. $\oplus$ is the symbol for XOR.

- $Q = h_1(i.d.)$
- random $\rho \in \{0,1\}^n$
- integer $r = h_3(\rho, m)$
- $g = e_l(Q, sP)^r = e_l(Q, P)^{rs}$
- $V = \rho \oplus h_2(g)$
- $W = m \oplus h_4(\rho)$
- $U = rP$
- Transmits the ciphertext (U, V, W)

Decrypt In order for Alice to decrypt the message, (U, V, W), which she has received from Bob, she performs the steps below. Her private key is sQ.

- if $U \notin E$ then reject ciphertext
- $g = e_l(sQ, U) = e_l(sQ, rP) = e_l(Q, P)^{rs}$
- $\rho = V \oplus h_2(g)$
- $m = W \oplus h_4(\rho)$

- $r = h_3(\rho, m)$
- if $U \neq rP$ then reject ciphertext

The security of the scheme is based on the BDHP as the adversary knows Q which is a point on the curve and so can be written as $Q = xP$. It also knows $U = rP$ and sP it cannot be assumed that the domain parameters are a secret. Therefore it has to attempt to calculate $e_l(P, P)^{xrs}$ which it should not be able to do.

Over the curve $E(GF(2^{283}))$, U is 284 bits using point compression. V and W are the same length as the message, which could be an AES key, the smallest value that this can be is 128 (see Table 4.4.3). Therefore the total length of the ciphertext is 540 bits.

4.6.2 Identity-Based Signature

The Cha and Cheon Identity-Based Signature (IBS) scheme [17] can be used to digitally sign the message. This uses the same public domain parameter and private key as the Boneh-Franklin scheme with an additional hash function, $h_5 : \{0,1\}^* \times \mathbb{G}_1 \rightarrow \mathbb{Z}_l^*$. In place of encrypt and decrypt, there are two new algorithms; sign and verify.

Sign
- pick random $t \in \mathbb{Z}_l^*$
- $U = th_1(i.d.) = tQ$ where Q is the public key of the signing entity
- $h = h_5(m, U)$
- $V = (t + h)sQ$ where sQ is the private key of the signing entity
- Transmit message, m, and signature (U, V)

Verify
- $h = h_5(m, U)$
- $Q = h_1(i.d.)$
- The signature is valid if $e_l(sP, U + hQ) = e_l(P, V)$

This is because

$$e_l(sP, U + hQ) = e_l(sP, (t + h)Q) = e_l(P, (t + h)Q)^s,$$

$$e_l(P, V) = e_l(P, (t + h)sQ) = e_l(P, (t + h)Q)^s.$$

The signature U, V will be 568 bits using point compression over the curve $E(GF(2^{283}))$.

4.6.3 Identity-Based Signcryption

IBE could be used in conjunction with IBS to distribute symmetric keys, or small messages, in a WSN. Instead, it would be preferable to use signcryption [105] as this results in less computation and bandwidth usage. An efficient Identity-Based Signcryption (IBSC) scheme, that is a submission to the identity based public key cryptography IEEE P1363.3 standard, is the BLMQ scheme [7]. In the BLMQ scheme there are four algorithms; setup, keygen, sign/encrypt and decrypt/verify.

Setup As with the Boneh-Franklin scheme the domain parameters are generated by this algorithm. Points $Q \in \mathbb{G}_2$ and $P = \psi(Q) \in \mathbb{G}_1$ are chosen. A Tate pairing calculation is run to calculate $g = e_l(P, Q) \in \mu_l$. A master key, s, that is a random member of $\mathbb{Z}_l^*$ is chosen, which generates the system-wide public key, $Q_{pub} = sQ$. There are also two hash functions;

- $h_1 : \{0,1\}^* \rightarrow \mathbb{Z}_l$
- $h_2 : \{0,1\}^* \times \mu_l \rightarrow \mathbb{Z}_l$
- $h_2 : \mu_l \rightarrow \{0,1\}^n$

Keygen This maps an identity, ID, to a private key $S_{ID} = \frac{1}{h_1(ID)+s} Q$.

Sign/Encrypt In order for Bob to sign and encrypt a message, $m \in \{0,1\}^n$, which he sends to Alice, he performs the following steps. $\oplus$ is the symbol for XOR.

- Random $x \in \mathbb{Z}_l^*$ is chosen
- $r = g^x$
- $c = m \oplus h_3(r)$
- $h = h_2(m, r)$
- $S = (x + h)\psi(S_{ID_A})$
- $T = x(h_1(ID_B)P + \psi(Q_{pub}))$

- Transmits the ciphertext (c, S, T)

Verify/Decrypt In order for Alice to decrypt the message, (c, S, T), which she has received from Bob, she performs the steps below.

- $r = e_l(T, S_{ID_B})$
- $m = c \oplus h_3(r)$
- $h = h_2(m, r)$
- if $r \neq e_l(S, h_1(ID_A)Q + Q_{PUB})g^{-h}$ then reject ciphertext

The ciphertext, c, S, T, will be 696 bits using point compression over the curve $E(GF(2^{283}))$.. This is smaller than would be the result if a signed message is encrypted using IBS and IBE. Therefore BLMQ IBSC scheme would be the appropriate choice for transmitting a symmetric key or a small message in a WSN. In the same paper they also introduce an efficient IBS scheme. The signature on a message m is (h, S) from above, and the message is only accepted if $h = h_2(m, e_l(S, h_1(ID)Q + Q_{PUB})g^{-h})$

4.6.4 IBC in a WSN

With IBC the wireless sensor node's identity can be used as the public key and hence there is no need for a certificate to bind a wireless sensor node's identity to its public key. The base station used to program the wireless sensor nodes could play the role of the KGC in the network (see Figure 4.5). The steps required for communication from Bob to Alice are given in Algorithm 5.

Algorithm 5 IBC Data Transfer

1: Bob encrypts message using Alice's identity.
2: Bob signs encrypted message using its private key.
3: Bob sends encrypted message and digital signature to Alice.
4: Alice decrypts message using private key.
5: Alice verifies digital signature of message using Bob's public key.

In comparison to the scheme presented in Section 4.5 there is only one radio transmission required, making this approach much more suitable for low energy WSNs. From

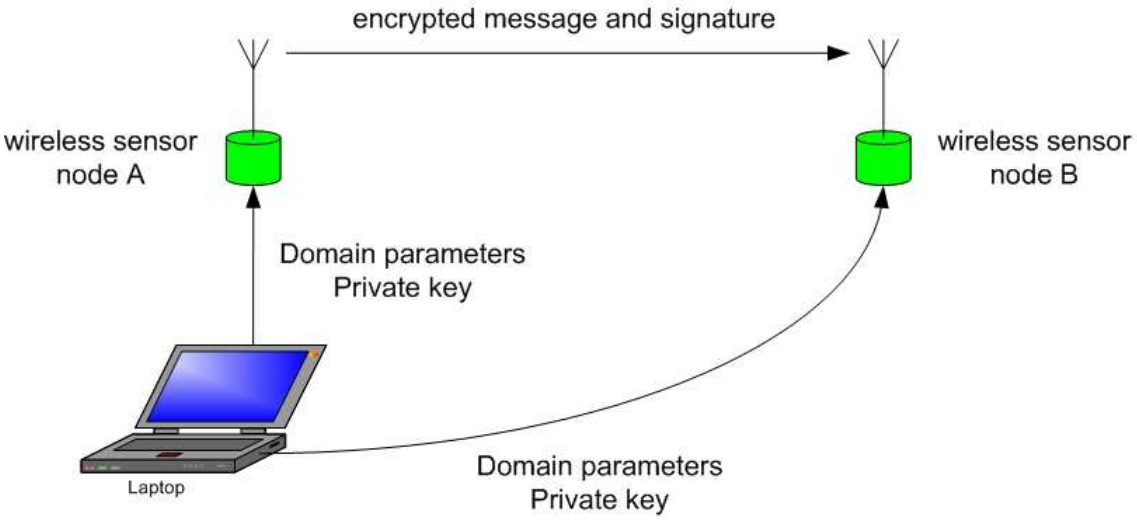

Figure 4.5: IBC deployed in a WSN

Section 2.3 on page 17 it can be seen that the networks will be organised in a multi-hop fashion. This leads to the conclusion that an IBC scheme will greatly reduce the energy dissipation in the network. For example, when Alice wants to encrypt a message to Bob who is ten hops away using a traditional PKI, then Bob's digital signature would have to be relayed ten times to Alice via the intermediate wireless sensor nodes, which is not the case in IBC.

The fact that only wireless sensor nodes that are programmed by the KGC will have valid private keys guarantees network access control. As even if an adversary asks for a message to be encrypted to a certain identity, it cannot decrypt the message as it does not have the private key. Also, it cannot inject a message into the network, as it does not have a private key to generate a valid signature.

This system is easily scalable, as new wireless sensor nodes only have to be programmed with the domain parameters and a private key by the KGC before deployment.

4.6.5 Identity-Based Non-Interactive Key Distribution Scheme

Instead of using IBSC to distribute a symmetric session key; the bilinear pairing could be used generate the key between the wireless sensor nodes. SOK identity-based Non-Interactive Key Distribution Scheme (NIKDS) was proposed by Sakai, Ohgishi and Kasahara [85] and can be implemented using the Tate pairing. If given $h_1 : \{0,1\}^* \rightarrow \mathbb{G}_1$ and

two devices with identity A and B respectively. Then $Q_A = h_1(A)$ and $Q_B = h_1(B)$ where $Q_A, Q_B \in \mathbb{G}_1$. The wireless sensor nodes have their private key sQ_A and sQ_B placed on them by the KGC. The symmetric key, k_{AB}, can be calculated by both parties as;

$$k_{AB} = e_l(sQ_A, Q_B) = e_l(Q_A, Q_B)^s = e_l(Q_A, sQ_B) \tag{4.7}$$

4.6.6 Conclusion

From the above it can be seen that an asymmetric encryption protocol is desirable for distributing symmetric keys in a WSN. Of the asymmetric approaches considered, the SOK identity-based NIKDS would be the best choice as it does not require any bits to be transmitted.

4.7 Protocol

The protocol outlined here, for implementing security in a WSN, is designed for a static network, and uses SOK identity-based NIKDS and BMLQ IBS. The environmental threat detection in Section 1.2.3 is the target application. In this case the wireless sensor nodes that are detecting the chemical reagents are given a fixed position. Techniques, such as those outlined in Section 2.4, are used so that every wireless sensor node can work out its position. End users have to be able to easily extract data from the network and this can be achieved using a Personal Digital Assistant (PDA) type devices. The protocol uses identity-based NIKDSand IBS as a method for distributing symmetric keys, and also to allow devices access to the WSN.

There are four distinct stages to this protocol; prior to deployment, deployment, re-keying and wireless sensor node addition, and data extraction. Each one of these stages are outlined below. The wireless sensor nodes themselves are not protected by tamper resistant hardware as this would increase their cost. Therefore it is possible that data and keying material on the devices can be extracted. It is assumed that some of the devices will be compromised in this way, but this number should be less than 10% of the total number

in the WSN. Also the KGC, which programs the devices, and the PDAs, which extract information form the WSN, are secure. The users of the network would be able to find out if a PDA is lost and hence exclude that particular device from any more communication. In the following, S and V indicate IBS signature generation and verification.

Prior to deployment This part of the protocol is concerned with distributing the domain parameters and private keys to the wireless sensor nodes. The elliptic curve and Galois fields being used are hard coded on the device. For SOK NIKDS the devices have to be able to calculate $K_{AB} = e_l(sQ_A, h_3(B))$. For BLMQ IBS they have to be able to generate a signature, (h, S), and verify a signature, $h = h_2(m, e_l(S, h_1(ID)Q + Q_{PUB})g^{-h})$ (see Section 4.6.3). Therefore the parameters that are placed on the devices are

- $(h_1, h_2, Q, Q_{PUB}, g) \rightarrow N_X$
- $Q_{KGC}, Q_X, sQ_X \rightarrow N_X$

where a generic wireless sensor node is given the identity N_X and has public key $Q_X = h_3(N_X)$ and private key sQ_X. Q_{KGC} is the public key of the KGC. Instead of placing h_3 on the device, the hash function is carried out by the programming device and the point on the curve to which an identity equates to is placed on the wireless sensor node. For the rest of this section Q_* represents the identity of the wireless sensor node and also its public key.

Deployment During this phase symmetric keys are set up between neighbouring wireless sensor nodes in a pairwise fashion. The wireless sensor nodes would transmit a small signed message to every device in radio range at time T_1. This would means that devices that can generate a valid signature are permitted to join the network. They would then generate a pairwise symmetric key, K_{AB}. The wireless sensor nodes maintain a list of authenticated devices in radio range.

- $Q_A \rightarrow Q_B :$ $\quad m||S(m)$
- $Q_B :$ $\quad V(m||S(m))$

- Q_B : $K_{AB} = e_l(sQ_A, Q_B)$.
- Maintain list of wireless sensor nodes in radio range i.e. $\mathbb{C}_B = \{Q_A, Q_C, Q_D\}$

Where K_{AB} is a shared symmetric key between Q_A and Q_B, S represents signing and V represents verification.

wireless sensor node addition At time T_3 extra devices may be added to the WSN. At a previous time, T_2, the KGC will broadcast though the network the identity of the wireless sensor nodes to be added i.e. $\mathbb{E}_{KGC} = \{Q_O, Q_P, Q_Q\}$. The identities of these devices, along with a time-stamp, are signed by the KGC. It does this in order to authenticate these identities and prevent the same message being replayed by an adversary in the future, that may have extracted private keying material from one of the wireless sensor nodes in $\mathbb{E}_{KGC}$.

- $Q_{KGC} \rightarrow Q_X$: $Q_O||Q_P||Q_Q||T_2||(S(Q_O||Q_P||Q_Q||T_2))$
- Q_X : $V(Q_O||Q_P||Q_Q||T_2||(S_{sQ_{KGC}}(Q_O||Q_P||Q_Q||T_2))$
- $Q_X \rightarrow Q_A$: $m||S(m)$
- Q_A : $V(m||S(m))$
- Q_A : $K_{AX} = e_l(sQ_A, Q_X)$.
- Q_A : If $Q_X \notin \mathbb{C}_A$ or $\mathbb{E}_{KGC}$ then reject Q_X else $Q_X \in \mathbb{C}_A$

Data extraction In the environmental monitoring scenario, presented in Section 1.2.3, it is envisaged that the WSN itself would be static, but that the entities extracting data from the network are mobile. For example, they could be a policeman who uses a PDA to extract information from the network. The PDA in this case will be programmed with the same domain parameters as the wireless sensor nodes. Only wireless sensor nodes authorised by the KGC can join the network; hence the KGC needs to send a packet that contains the identity of Q_{PDA} and is signed by its private key.

- $Q_{KGC} \rightarrow Q_X$: $Q_{PDA}||S(Q_{PDA})$

- Q_X : $V(Q_{PDA}||S(Q_{PDA}))$
- Q_X : $K_{XPDA} = e_l(sQ_X, Q_{PDA})$.

When a PDA requests a reading, from a certain geographical area, it will diffuse this request through the network. As its identity Q_{PDA} has already been broadcast to the network as a valid identity, then the wireless sensor node that is elected to send data back to the PDA can do so using AES. This message and the identity of the source is further encrypted by the local symmetric key and forwarded toward the extraction point, which is also known as a sink. It may be symmetrically encrypted and decrypted many times before the destination is reached.

4.8 Security of the Protocol

The security of this protocol is based on the fact that the WSN, or the portion of it that takes the measurements, is static. The wireless sensor nodes build up tables of adjacent wireless sensor nodes with which they can communicate. Upon capture of a device and the extraction of its keying material then the adversary will only have access to a very small number of nodes' data. The utility of a WSN is based on its massively distributed nature and the fact that many wireless sensor nodes collaborate to give an accurate reading of what is being measured. In this case, the fact that only several wireless sensor node's data have been extracted will result in an inaccurate measurement. Another major advantage to re-keying the WSN regularly is that if a wireless sensor node is captured and the symmetric key extracted then the adversary will only have access to the previous communications that were encrypted with that particular key. All previous communication is still secure.

Various different attacks are discussed on the WSN in the following section. They take the form of erroneous data insertion, routing disruption or attacks to deplete the energy source of the device.

4.8.1 Erroneous Data Insertion

A malicious node may introduce erroneous data into the network, but if it is not collaborated by another node then the data is disregarded. For example one wireless sensor node might indicate the presence of a chemical reagent but if other devices, with which it shares pairwise keys, do not detect the chemical present then the reading is not accepted as valid. Mechanisms could be put in place such that a number of invalid readings being detected by two or more wireless sensor nodes will lead to the offending device being excluded from the WSN.

There is also an attack whereby a compromised wireless sensor node, that is elected to communicate with the sink, sends erroneous data to this sink. An attack of this nature is difficult to defend against. One approach is for the wireless sensor nodes in a geographical area to agree upon a clusterhead wireless sensor node that is responsible for communication with the sink. This clusterhead could be assigned in a round robin manner, so that in a group of ten devices they only have one turn every ten transmissions. The identity of the clusterhead is forwarded, with the message, to wireless sensor nodes along the path to the PDA. If, along the path, the wireless sensor nodes notice that this identity was the clusterhead less than ten transmissions ago then it will drop the packet. A malicious device could try and forward a message to the sink and spoof a valid clusterhead identity but this is not feasible, as it will not have the symmetric key associated with the spoofed identity.

4.8.2 Sinkhole Attack

In a sinkhole attack, a compromised device advertises a high quality route to a data extraction point, when it is not near one. This causes data to be routed to this malicious wireless sensor node, which can then drop the packets. As the wireless sensor nodes are aware of their position, this attack can be easily countered. If the device injects false routing information, to say that it is close to a distant area of the network, then as the wireless sensor nodes in the next hop are aware of their own position they will know that this could not be the case, and drop the packet. Also, if routing information is replayed from another section of the network then the receiving device will ignore the communication as the

device from which the routing information originally is not a member of $\mathbb{C}_X$, where X is the wireless sensor node's identity.

4.8.3 Wormhole Attack

The wormhole attack [46] is where two devices, that are not wireless sensor nodes, and are geographically distant, conspire with each other to provide a low latency, undetectable (to the other devices in the WSN) route between them. For example the PDA or sink might request information from a certain location in the WSN known as a source. This request is distributed in a multi-hop fashion through the network. If the area of interest is ten hops away (see Figure 4.6) and there is a wormhole between device two and device seven then wireless sensor node eight believes that it is two hops from the sink. All communication between the source and sink would go through this wormhole as it appears to be a short path to the sink. The adversary could exploit this traffic to drop packets. The protocol

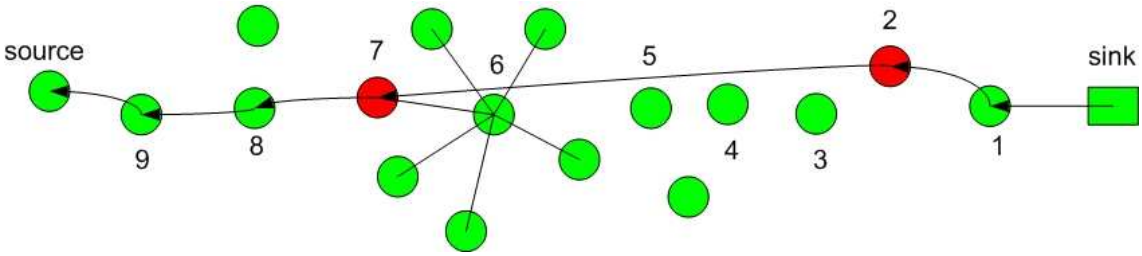

Figure 4.6: A representative wormhole attack: there is a wormhole between the two devices highlighted in red

defends against this attack as wireless sensor node eight will not accept a message from wireless sensor node one, as it is not in the list of devices that it can communicate with.

4.8.4 Sybil Attack

Sybil attacks [26] can be mounted by compromised devices. In this attack the wireless sensor nodes present multiple identities to neighbouring devices in order to disrupt routing, or provide multiple readings to the network to make the local aggregated data value erroneous. Under the protocol presented, this attack is no longer feasible, as during normal operation the wireless sensor nodes only accept packets from their neighbours in

$\mathbb{C}_X$. During the wireless sensor node addition phase they will only accept communication from devices in $\mathbb{C}_X$ or $\mathbb{E}_{KGC}$. Even if the Sybil identity presented is a member of $\mathbb{E}_{KGC}$ it cannot join the network as it does not have a private key for network access.

4.8.5 Identity Replication Attack

Unlike the Sybil attack, the identity replication attack [70] is based upon giving the same identity to different physical devices. This attack can be mounted because in a WSN there is no way to know that a wireless sensor node is compromised. If this device is cloned and placed in different parts of the network with the intention of disrupting the routing protocols then this attack can be overcome with the security protocol. Since the wireless sensor nodes are only allowed to communicate with other devices that are a member of $\mathbb{C}_X$ or $\mathbb{E}_{KGC}$. Hence, if a device were moved to another part of the network it would not be able to join the WSN at that point. A variation on this attack would be for many cloned devices to be replaced in the same location from which the wireless sensor node was originally taken. The aim of this attack might be to use up all the available bandwidth or inject many erroneous data readings into the network. The medium access control function of the link layer will prevent this type of attack as the wireless sensor nodes could be given access to the medium in a round robin fashion.

4.8.6 Energy Depletion Attack

Adversaries could mount energy depletion attacks. The aim of this attack would be to deplete the battery of the wireless sensor nodes in order to destroy the network. Attacks of this nature can take various forms; they can be carried out by devices that have been compromised but also by entities claiming to be a member of the WSN. As during normal operation the wireless sensor nodes only accept data from their immediate neighbours, these attacks would have to be carried out during the wireless sensor node addition phase or when the devices communicate with a sink.

During the wireless sensor node addition phase of the protocol an external adversary, which is a device that does not have valid keying material, could attempt to access the

network. It cannot join the WSN as it does not have a valid private key, but the aim of the attack is to make the wireless sensor nodes perform this verification step and so waste energy. Also a compromised device that mounts this attack cannot join the network as it is only a member of $\mathbb{C}_X$ at the geographical location from where it was taken. This attack cannot be defended against but systems can be put in place to limit its damage. As the wireless sensor nodes knows the number of devices in $\mathbb{E}_{KGC}$, they will only be expecting to receive a maximum number of transmissions, anymore and its knows that there is an attack taking place. By setting the maximum number of transmissions that a device can receive to be the number in $\mathbb{C}_X$, in this phase of the protocol, then this should prevent the attack from succeeding.

In normal operation all the wireless sensor nodes receive their data from devices that are a member of their group, i.e. $Q_X \in \mathbb{C}_X$. There are also occasions that they have to interpret a broadcast message from outside the static WSN, such as from the KGC or from a sink. This is an obvious target for mounting an energy dissipation attack. An adversary could pretend to be the KGC and ask for several wireless sensor node identities to be added to the WSN. Of course, this would be refused as it would be able to sign the message, but it would lead to a dissipation of the battery on the devices performing the verification algorithm. One way to limit the damage of this attack would be to only allow a small number of the wireless sensor nodes to access data from outside the WSN at any one time, and also the addition of devices can only occur at predetermined times. Simulation is required to ascertain the smallest percentage of wireless sensor nodes that must be available to accept external data.

4.9 Software profile of the Protocol

In this thesis, in order to evaluate whether a protocol based on SOK identity-based NIKDS and BMLQ IBSis an appropriate choice for a WSN, the most computationally demanding components are implemented in software using the curve $E(GF(2^{283})) = y^2 + y \equiv x^3 + x + 1$ and the Miracl library [90]. The components that are profiled are exponentiation, point multiplication and the Tate pairing. In Table 4.9 the Tate pairing contribution

to the total is counted twice as it is used in signature verification and symmetric key generation. The target device is the ARM920T [5] as this is a device that could be used on a wireless sensor node. The platform used to obtain these measurements is shown in Figure 4.7. The code used to implement the scheme was compiled for the ARM using

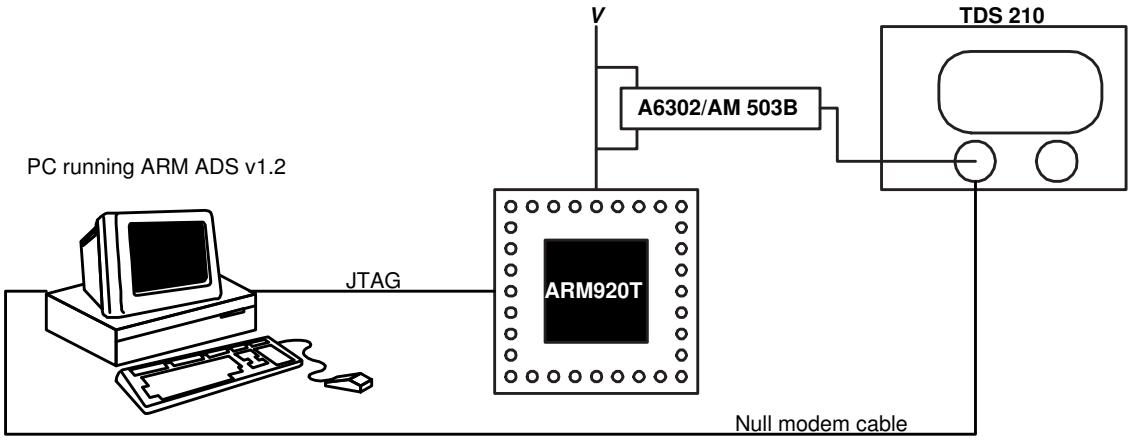

Figure 4.7: Experimental setup for measuring time and energy of the ibe protocol [57]

the ARM Development Suite (ADS) v1.2. As well as generating an executable that can be downloaded to the ARM using the JTAG inputs, it also gives timing figures for these executables. The wire into the core is split in two and one branch is wrapped around the Tektronix A6302 current probe amplifier that is connected to the Tektronix TDS 210 oscilloscope. The C code is written so that the code for which the current is to be measured is a large loop that triggers the oscilloscope every iteration through the GPIO. The output from the oscilloscope is captured for analysis on a PC. The instantaneous current, $i(t)$, is captured and must be multiplied by the voltage, V_{DD}, and integrated over the program execution time, T, to obtain the energy, E, as given in Equation (4.8). The voltage over which the measurements are taken is 2.5V and the frequency is 140MHz.

$$E = V_{DD} \int_0^T i(t) \mathrm{d}t \tag{4.8}$$

The total energy dissipated is $35.4mJ$ and the power consumed is $0.05W$ at 140 MHz. The time required to run the protocol algorithm at 200 MHz is $444.5ms$. Due to the nature of the experimental setup these figures are a lower bound of the energy dissipated by these component parts, as it has been assumed that the current through both branches going into

code	algorithm	time (ms)	energy (mJ)
tate ($\times 2$)	$e_l(P,Q)$	177.1	14.1
power	g^x	39.8	3.2
mult	rQ	50.6	4.0
	Total	444.5	35.4

Table 4.2: Timing and energy figures for main components of the protocol

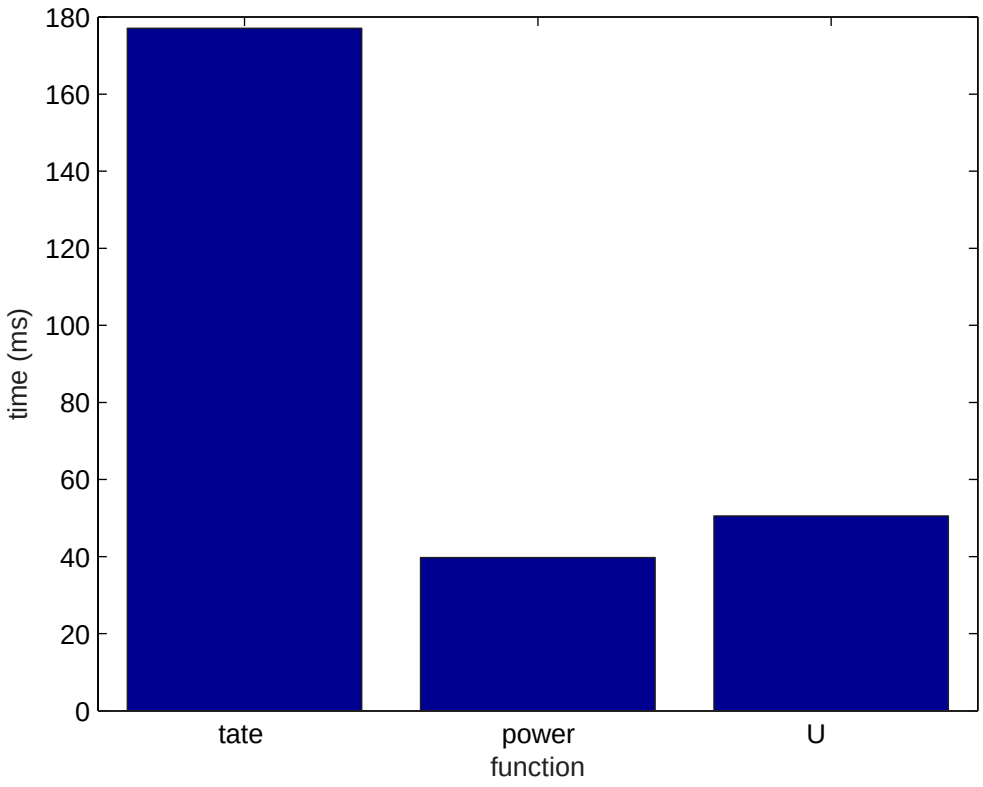

Figure 4.8: Time required for main components of the protocol

the device is equal when this cannot be the case. The current going through the branch that is measured would be smaller due to greater area and length of the wire, thus the energy and power figures should be higher than provided. The energy measurements should be taken at 200 MHz as this will be he clock speed of the final system. This could not be achieved as the fastest clock speed that the board can run at is 140 MHz. It is can be seen that a software implementation requires too much energy and its latency is unacceptably large for implementation on a wireless sensor node, therefore a hardware implementation of this scheme should be investigated.

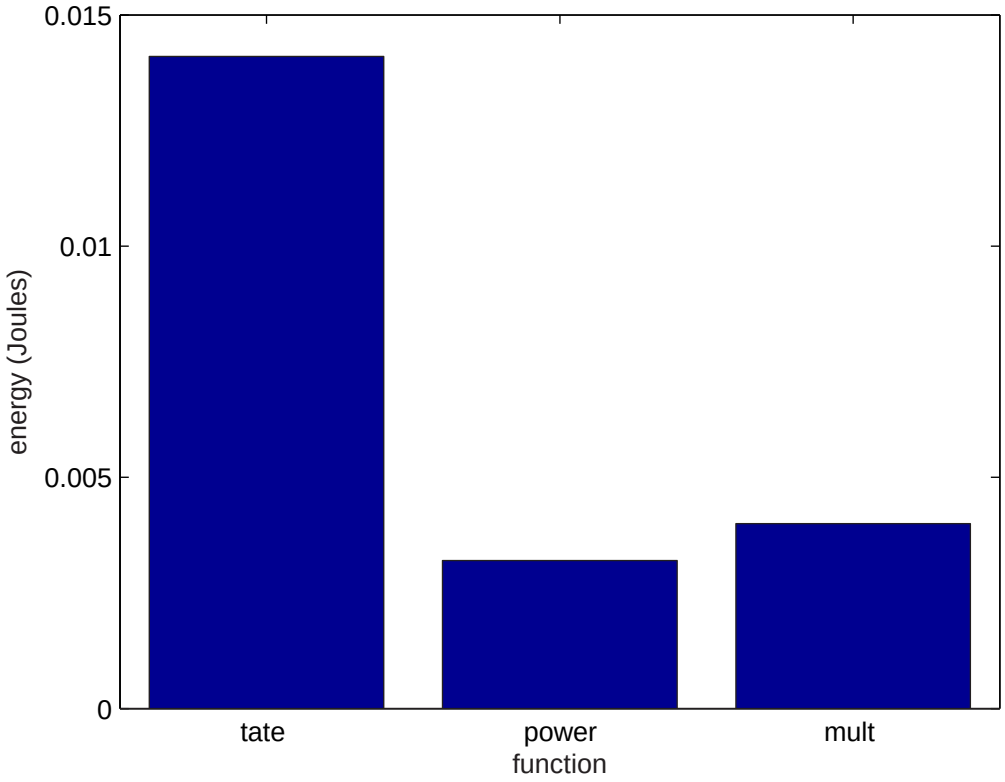

Figure 4.9: Energy required for main components of the protocol

From analysis of the figures and the table in this section it is clear that the Tate pairing calculation is the most computationally demanding component of the protocol. It requires $14.1mJ$ to run which is considerable when the total energy budget of the wireless sensor nodes is of the order of $1000J$. A key design goal of wireless sensor nodes is that they operate on a low duty cycle and the fact that the Tate pairing takes $177.1ms$ is counter to this goal. A hardware implementation of the Tate pairing is therefore merited, and it will reduce the time it requires and also the energy it dissipates.

4.10 Summary of Contributions

In this chapter a discussion of the efficacy of various approaches to providing security in a WSN is presented. It is concluded that a symmetric scheme is the appropriate choice for communication due to the limited energy resource of the network, though the deployment of symmetric keys in the WSN is a challenging problem. Pre-deployed keys are not

appropriate for reasons of security and also, in the case of pairwise keys, their failure to scale. It is argued that the symmetric keys should be distributed through the WSN using an asymmetric protocol.

As one of the criteria for adopting a particular protocol for security is that it minimises energy dissipation it is evident that, of the asymmetric schemes available, those based on an elliptic curve are the best choice. This is because the key sizes for these schemes are four to six times smaller than for example RSA, and hence the radio is required to transmit fewer bits. It has been shown that a IBC scheme is preferable to a ECC one because, it does not require the transmission of digital certificates and there is a simple non-interactive process for generating symmetric keys.

A protocol that uses BMLQ IBS and SOK identity-based NIKDS is presented. This protocol performs a number of functions; authenticates network access, distributes local symmetric keys and provides a mechanism for the WSN to communicate securely with a sink. Standard attacks on a WSN are also described. The fact that it is assumed that wireless sensor nodes know their own position, and also the nature of the protocol, enables the WSN to defend itself against these attacks.

This work is similar to the approach presented in [104], in that it also uses location in its security mechanisms. The security of their system is based on the fact that a network master secret, that is used to generate location based keys, is kept secret for a minimum time. This is the time that it is believed that an adversary would require to access this key if in control of the wireless sensor node. When the wireless sensor nodes have all calculated their location based keys then this network master secret is securely erased. Additional devices added to the network will require access to this network master secret. This scheme's security depends upon keeping this network master secret secure, whereas the one presented in this chapter assumes that the adversary can access a proportion of the wireless sensor nodes private keying material and will still function as a secure WSN.

The proposed protocol has not been validated on either a wireless sensor node system or a simulation environment. This was not done at this stage as it unclear whether or not the protocol could be implemented on a device that is very constrained in terms of its

energy resources. In order to investigate whether or not the protocol can be implemented in a WSN the most computationally parts of the protocol are profiled in software. A characteristic two version is profiled, targeting an ARM920T, in terms of time and energy. The figures indicate that a software implementation will be too costly in terms of energy and time. Therefore a hardware implementation is warranted. From the profiling results it can be seen that the Tate pairing should be implemented in dedicated hardware in order to reduce the running time and energy required for the protocol. In the following chapters a co-processor for the Tate pairing is designed, implemented and evaluated to ascertain if the scheme presented in this chapter could be implemented on a wireless sensor node.

CHAPTER 5
Arithmetic Units

5.1 Introduction

From the discussion in Chapter 4 it is clear that the Tate pairing contributes over 66% to the total energy and run-time of the protocol based on IBC. For this reason it is a suitable candidate for hardware acceleration. A hardware accelerator will be implemented as a peripheral device on an ARM system [5]. The reasons a 32-bit processor was decided upon is that an algorithm run on a 32-bit machine will be much faster than one run on an 8-bit machine, such as the Atmel ATmega 128L used for MICA2. Therefore a lower voltage can be used, and as energy used is proportional to the square of voltage, there is a considerable saving in energy.

5.2 Design Considerations

The hardware accelerator will be implemented in CMOS technology and it is important that due consideration is taken of the design goals which can be summarised as low energy and low cost.

5.2.1 Circuit Energy Dissipation

Before a discussion on how to reduce digital energy dissipation, it is instructive to review the main areas of energy consumption in CMOS digital circuitry. For the total average energy, E_{avg}, there are three main components;

$$E_{avg} = E_{dynamic} + E_{static} \tag{5.1}$$

where $E_{dynamic}$ is dynamic energy, E_{short} is short circuit energy and E_{static} is static energy. These components will be described in the following sections.

5.2.1.1 Dynamic Energy Consumption

In CMOS circuits the mechanism that consumes the most energy is charging and discharging of the capacitive load [77]. A device such as an inverter can be modelled as two resisters, a switch and a capacitor (see Figure 5.1). This capacitance is a result of the parasitic capacitance of the device itself, the interconnect and the load that the device is connected to.

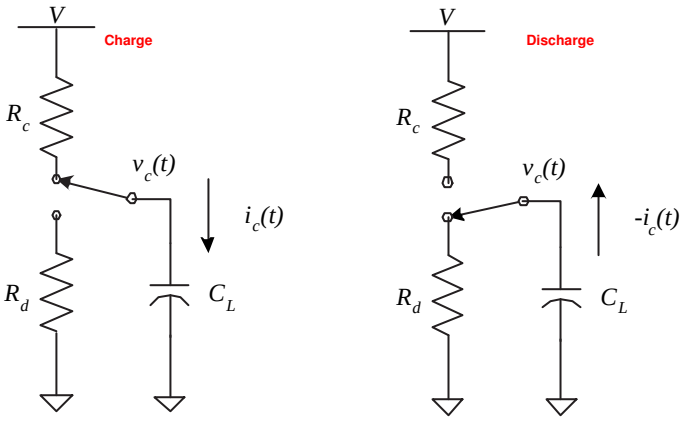

Figure 5.1: Dynamic Charging and Discharging a Capacitive Load [57]

The capacitance, C_L, is charged when the p-channel Metal Oxide Silicon (PMOS) device is on, and the resistance of the PMOS device is modelled by R_c. The current charging the capacitive load is i_c. As the capacitor is charged the energy used $E_{V_{DD}}$ is

given by,

$$\begin{aligned} E_{V_{DD}} &= \int_0^\infty V_{DD} i_c \mathrm{d}t \\ &= C_L V_{DD} \int_0^\infty \frac{\mathrm{d}v_c}{\mathrm{d}t} \mathrm{d}t \\ &= C_L V_{DD} \int_0^{V_{DD}} \mathrm{d}v_c \\ &= C_L V_{DD}^2 \end{aligned} \tag{5.2}$$

Half of this energy is stored in the capacitor, E_{cap}, and the rest is dissipated in the resistor R_c

$$\begin{aligned} E_{cap} &= \int_0^\infty v_c i_c \mathrm{d}t \\ &= C_L \int_0^\infty v_c \frac{\mathrm{d}v_c}{\mathrm{d}t} \mathrm{d}t \\ &= C_L \int_0^{V_{DD}} v_c \mathrm{d}v_c \\ &= \frac{1}{2} C_L V_{DD}^2 \end{aligned} \tag{5.3}$$

When the n-channel Metal Oxide Silicon (NMOS) device is active for the logic transition from one to zero then the capacitor is discharged through the NMOS device whose resistance is given by R_d. The energy in the capacitor E_{cap} is dissipated as heat. Therefore in a charge / discharge cycle the total energy dissipated is $E_{V_{DD}}$.

If the probability that a node transitions from zero to one in a particular clock cycle is given by α, then the average energy dissipated by this node in a clock cycle is E_{node_dyn}.

$$E_{node\_dyn} = C_L V_{DD}^2 \alpha \tag{5.4}$$

For a complete circuit then the total dynamic energy consumed, E_{total_dyn} (5.5), is the sum of the energy consumed in the individual nodes multiplied by the number of clock cycles , N_{cycle}, required to carry out the required processing.

$$E_{total\_dyn} = \left(\sum_i C_i V_{DD}^2 \alpha_i \right) N_{cycle} \tag{5.5}$$

The total power consumed for a certain frequency, f, is,

$$P_{total\_dyn} = \left(\sum_i C_i V_{DD}^2 \alpha_i \right) N_{cycle} f \tag{5.6}$$

From an analysis of Equation (5.5) it can seen that reducing the voltage will have the greatest effect on E_{total_dyn}. If voltage, V_{DD}, is reduced then this increases the delay of the device, T (see Equation (5.7)), and this might result in the latency of the design being violated. To fix this violation the techniques of parallelism or pipelining can be employed. In the case were the design is made more parallel this will lead to an increase in area, and hence capacitance, but as this is a linear term in equation (5.5) there is always a reduction in the overall energy used.

$$T = \frac{C_L}{\frac{\mu C_{ox}}{2} \frac{W}{L}} \frac{V_{DD}}{(V_{DD} - V_t)^2} \tag{5.7}$$

Other approaches to reduce E_{total_dyn} is to gate signals to a known value when they are not required. This has the effect to making α equal to zero. The gating technique can be utilised in the datapath or more commonly the clock is gated to logic zero when a particular module is not required. As the clock tree can consume 30% of total power [69] this technique can lead to considerable savings. Glitching of a circuit is undesirable, but it is unavoidable, it should be reduced as much as possible by using balanced paths in the datapath. Finally, a reduction in the number of clock cycles an algorithm takes to execute will reduce the energy consumed.

5.2.1.2 Short-Circuit Energy Consumption

Due to the finite slope of the input waveform driving a CMOS circuit there is a point at which both the NMOS and PMOS transistors are on. This means that there is a direct path between V_{DD} and ground - i.e. a short circuit. The short circuit current is represented in Figure 5.2. Equation (5.8) for the the average energy dissipated by this node in a clock

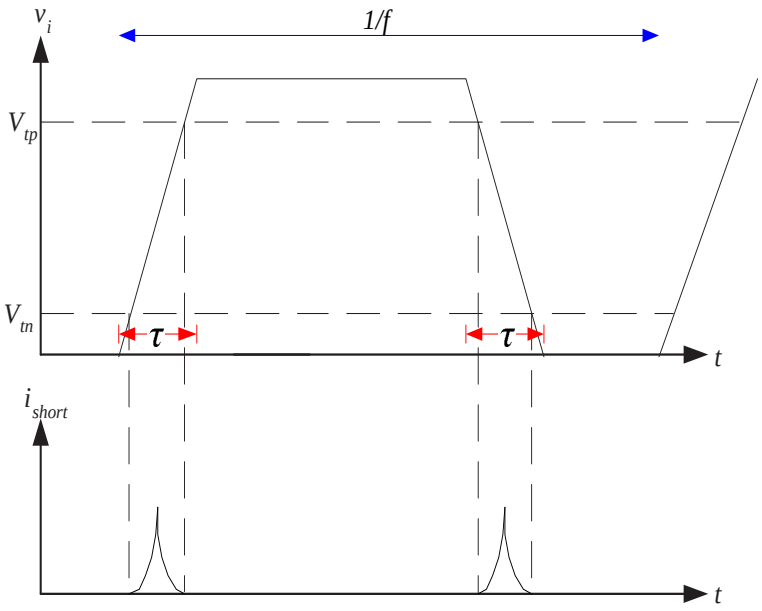

Figure 5.2: CMOS Inverter Short-Circuit Current [57]

cycle is derived from Figure 5.2 and is given by,

$$\begin{aligned} E_{node\_sho} &= \left(V_{DD} \frac{I_{peak}\tau}{2} + V_{DD} \frac{I_{peak}\tau}{2} \right) \alpha \\ &= V_{DD} I_{peak} \tau \alpha \end{aligned} \tag{5.8}$$

where the probability that a node transitions from zero to one in a particular clock cycle is given by α, τ is the time both transistors are on and I_{peak} is the peak short circuit current.

As in section 5.2.1.1 the total energy consumed, E_{total_sho} (5.9), is the sum of the individual nodes multiplied by the number of clock cycles , N_{cycle}, required to carry out the required processing.

$$E_{total\_sho} = \left(\sum_i V_{DD} I_{peak} \tau \alpha_i \right) N_{cycle} \tag{5.9}$$

Short circuit power can be kept within the bound of 10% of the dynamic power if the rise and fall times of inputs and outputs are the same [77]. Techniques for reducing dynamic power will also help in reducing the short circuit power, as will matching input signal rise and output signal fall times.

5.2.1.3 Static Energy Consumption

When a CMOS device is in its steady state it ideally should not be conducting current, though in practice this is not the case. There are two main sources of leakage current; reverse biased diode junction current and sub-threshold current. In a Taiwan Semiconductor Manufacturing Company (TSMC) [98] 65nm low power process employed in this work the leakage power is two orders of magnitude less than the dynamic power components. Therefore, static energy consumption is a secondary concern of the work. But for completeness we have reviewed static power consumption.

Reverse biased diode junction current arises between the source or drain and the substrate (see figure 5.3). The current, due to this effect, can be modelled using the following

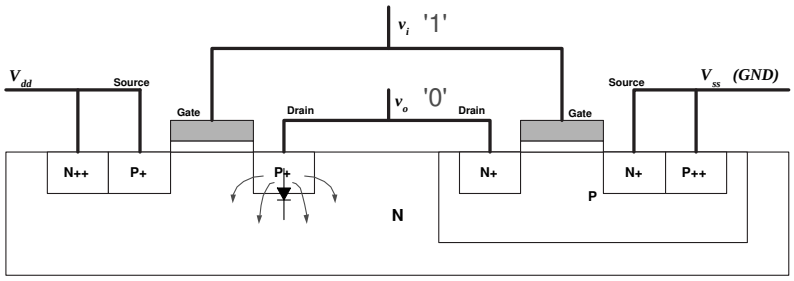

Figure 5.3: Diode Leakage Current in a CMOS Inverter [57]

formula for a reverse biased PN-junction,

$$I_{leakage\_jun} = I_s \left(e^{\frac{V_{DD}}{V_T}} - 1 \right) \tag{5.10}$$

where V_{DD} is the voltage across the junction, V_T is the thermal voltage and I_s is the saturation current [77].

The other source of leakage current is due to the current that flows when the voltage from gate to source, V_{GS}, is less than the threshold voltage, V_t, and is given by,

$$I_{leakage\_sub} = I_0 W e^{\left(\frac{-V_t}{nV_T}\right)} \left(1 - e^{-\frac{V_{DD}}{V_T}} \right) \tag{5.11}$$

where I_0 and n are empirically derived, V_T is the thermal voltage, W is the gate width, V_{DD} is the supply voltage and V_t is the threshold voltage [56]. This component is gaining

in importance. For as V_{DD} is reduced with migration to smaller technologies then V_t is also reduced, and this leads to a higher sub-threshold current in the CMOS device.

The total energy dissipated by these two components is

$$E_{leakage} = 1/f(I_{leakage\_sub} + I_{leakage\_jun})V_{DD}N_{cycle} \tag{5.12}$$

where f is the frequency of operation of the circuit and N_{cycle} is the number of clock cycles required for the operation.

An effective method for reducing static power consumption is to use power gating; turning off portions of the design as and when they are not needed. This is achieved by pulling the power rails from V_{DD} and the ground to open by using large pull down / pull up transistors. Another technique is to use Multiple Threshold Complementary Metal Oxide Silicon (MTCMOS) whereby devices in the critical path have a low threshold voltage and hence are fast but consume a lot of static energy, whereas other devices not in the critical path have a larger threshold voltage [102].

5.2.2 Cost

As discussed above the cost of the wireless sensor node is of primary importance; it will be impossible to deploy large numbers of these devices in a network if they cost much more than one dollar and are expensive to deploy. Therefore the design, manufacturing and deployment costs must be kept to a minimum.

The cost of an Integrated Circuit (IC) [77] can be given by,

$$\text{cost per IC} = \text{variable cost per IC} + \frac{\text{fixed cost}}{\text{volume}} \tag{5.13}$$

The fixed cost or non-recurring engineering charge is the amount of capital required to design the IC. This can be broken down into the cost of the engineers, cost of the Computer Aided Design (CAD) software and ancillary costs such as offices etc. Of these costs the engineers contribute the most to the fixed cost, and so it is paramount to reduce the number of engineering man-hours required to complete a project. This could be achieved

by using productive engineers and employing the best CAD tools and flows to reduce the number of engineering man-hours required. The approach to IC design that is normally used is standard cell design as this allows an engineer to design and verify a very large IC that would not be possible using a full custom design flow. The fixed cost could also include the cost of buying soft or hard Intellectual Property (IP) from third parties that can be incorporated into the IC. From Equation (5.13) it can be seen that the fixed cost as a percentage of the overall IC cost reduces as the volume of production increases.

The variable cost can be seen as the cost of manufacturing the IC and is given by Equation (5.14) [77]

$$\text{variable cost} = \frac{\text{cost of die test } + \text{ cost of packaging} + \text{cost of die}}{\text{final test yield}} \tag{5.14}$$

The cost of testing the die is related to how long the die must remain on the tester machine. These machines can cost millions of dollars so a short test time means that more dies can be tested per minute and therefore the cost of the machine can be spread over more ICs. The way to reduce time on the tester is reduce the number of test patterns that are required to cover 95 - 98% of the stuck at faults. The economic benefits of design for test are discussed in [16].

A plastic package is more expensive than a ceramic package and can dissipate heat up to two watts [77]. Therefore as long as the heat dissipation of the circuit can be kept below this figure then a plastic package would suffice, and thus help to reduce the variable cost.

The cost of the die is a function of die area as given by Equation (5.15). Dies per wafer is the area of the wafer divided by the area of a die and die yield is the percentage of good dies on the wafer. This latter term is inversely proportional to the cube of die area, therefore the cost of the die is actually a function of the forth power of area.

$$\text{cost of die} = \frac{\text{cost of wafer}}{\text{dies per wafer } \times \text{ die yield}} \tag{5.15}$$

It should be noted that when designing a system there is a cost in area for algorithms that

are implemented in software as they will have to be stored in flash memory.

The final aspect of cost that needs to be addressed is the issue of ease of use for the people deploying the network. There would be not much point in designing a low cost sensor if it was costly to deploy. This means that the wireless sensor nodes should have hardware and software that enables them to establish a secure network without interaction with the person deploying the network. Of course, whoever is using the wireless sensor nodes will have to program their application onto the devices but this should be made as simple as possible.

5.2.3 Conclusion

For a design of this nature there is no real time constraint and so there can not be a latency that has to be met. Power is not important to this design, what is critical is the amount of energy that the device consumes. As a Lithium battery has a energy density of $2880J/cm^3$, which translates into $90\mu W/cm^3/year$ [83], then this figure of $2880J$ could be used as an energy constraint. It has been discussed previously that the device must operate on a low duty cycle to conserve energy; this requires a circuit that completes its operation quickly. At the same time the device should be as cheap as possible, and this would mean that the techniques of parallelism might not be appropriate as they will increase the area and hence the cost. Finally, when the circuit is operating it should consume as little energy as possible. These sometimes conflicting design goals of latency, area and energy are combined into a single metric known as area*energy*time which will be used to evaluate the circuits outlined in this chapter.

5.3 Architecture for wireless sensor node

Taking into account the preceding discussion a suitable architecture for a wireless sensor node is proposed in this thesis (see Figure 5.4). The key components to this SoC are the radio, sensor and digital computation sub-systems. The device as envisaged will have a fixed sensor attached to it and so its application areas will be restricted. It is unlikely that

the wireless sensor node's mission will be fixed; therefore a hardwired ASIC approach would not be justified. Instead the device should contain a processor so that it could be programmed for different applications whilst still using the same sensor. From the discus-

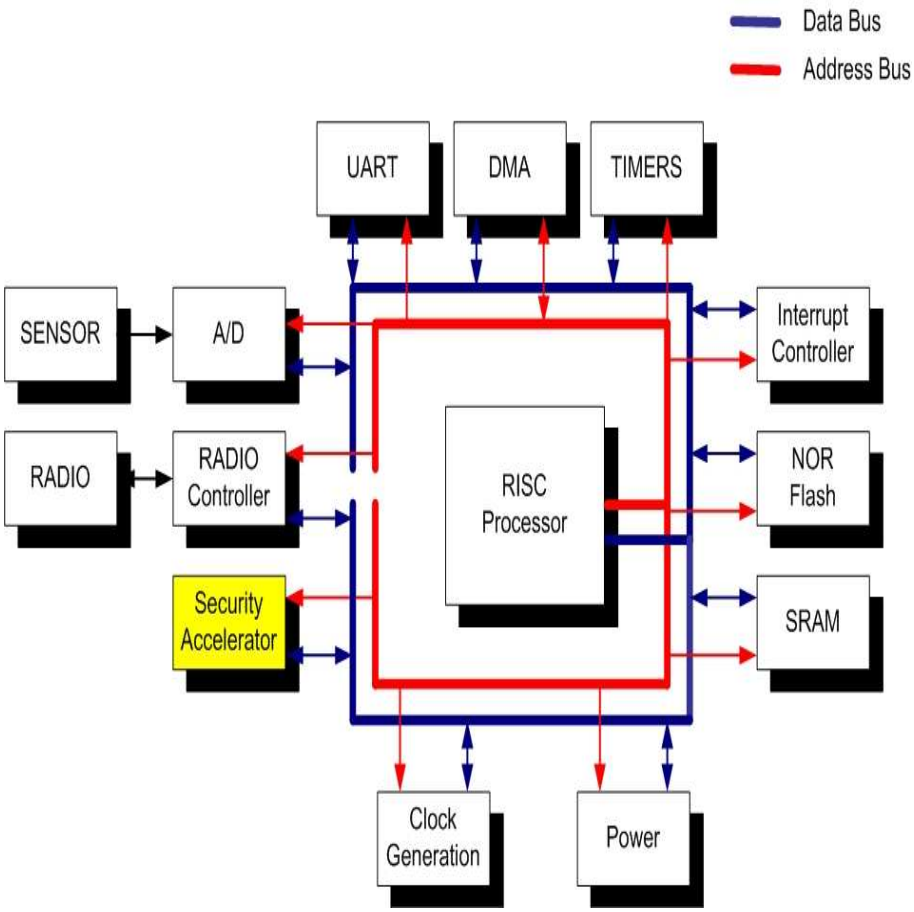

Figure 5.4: Architecture of a wireless sensor node

sion about cost (see section 5.2.2) the wireless sensor node should have as small an area as possible and still be a low energy / low power device. Therefore the number of peripherals and memory required by the device should be kept to a minimum. The design has to have persistent memory in the form of NOR flash for storing the code for running the wireless sensor node and also for storing measurements from the sensor. There is also SRAM for running the programs. A Universal Asynchronous Receiver Transmitter (UART) is provided for the purpose of programming the devices before deployment. In order to free up the CPU a DMA is incorporated in the design. This will allow direct access from the sensor and radio to the memory. As well as the standard block such as timers, clock extraction and interrupt controller; there is a power management block whose function is

to shut down blocks when they are not required. Finally, there is a security accelerator or crypto engine for implementing computationally demanding cryptography algorithms. It is in this block that the Tate pairing is instantiated.

5.4 Design Methodology

As a first step the algorithms are implemented in C++ using the Miracl library . This is in order to check that the algorithms are correct and to generate "golden" files that can be used in the verification of the hardware. An Algorithmic State Machine (ASM) diagram is written for the algorithm [14]. From this diagram a datapath circuit can be extracted. These first two diagrams are then used to write an ASM diagram for the control logic. The design can then be captured by Very High Speed Integrated Circuit Hardware Description Language (VHDL) using the diagram for the datapath and the ASM for the control logic.

The Register Transfer Level (RTL) code is verified using the Modelsim simulator with random data input and the "golden" output files generated by the C++ model. Synthesis and physical synthesis is performed using Synopsys Design Compiler and Physical Compiler, respectively. The target technology is a TSMC 65nm low power CMOS process [98]. Worst case operating conditions are used and these are; voltage 1.08V, process 1.000 and temperature $125°C$. It is synthesised for a clock of frequency of 125MHz, or a clock period of 8ns.

Following synthesis a Standard Parasitic Exchange Format (SPEF) file is extracted from Physical Compiler. This is converted into a Standard Delay Format (SDF) file using PrimeTime. A back-annotated VHDL netlist is then simulated using Modelsim to generate a Value Change Dump (VCD) file. The VHDL netlist, SPEF file and VCD file are used by PrimePower to arrive at a figure for the power consumption of the circuit. Clock power was not included in this figure. All of the above steps enable the area, latency, power and energy of the circuits to be measured.

5.5 Arithmetic Operations

All operations that are used for the various algorithms in the hardware accelerator take place in binary extension fields; either $GF(2^{283})$ or $GF(2^{283\times4})$. If the Tate pairing calculation is rewritten as in Algorithm 6, then the arithmetic operations that are required are addition, multiplication, and inversion in both fields. In addition, a circuit is required

Algorithm 6 η Algorithm [59, 6]

$P = (x_p, y_p), Q = (x_q, y_q) \quad : P, Q \in GF(2^{283})$
$C(x) = c_3x^3 + c_2x^2 + c_1x + c_0 = 1 \quad : C(x) \in GF(2^{283\times4})$
for $i = 1$ to 283 **do**
  $x_p = x_p^2$
  $y_p = y_p^2$
  $z = x_p + x_q$
  $w = z + x_px_q + y_p + y_q + 1$
  $C(x) = C(x)(w + zx + (z+1)x^2)$
  $x_q = x_q^{2^{283-1}}$
  $y_q = y_q^{2^{283-1}}$
end for
return $C(x) = C(x)^{2^{2\times283}-1}$

to perform the squaring and square root operations in $GF(2^{283})$, and exponentiation in $GF(2^{283\times4})$. How these functions are implemented in hardware is outlined below.

In the subfield $GF(2)$ addition is carried out using modulo two arithmetic, and hence can be performed in hardware using an XOR gate. Addition is equivalent to subtraction in $GF(2)$. Also, multiplication is performed using an AND gate in hardware.

The polynomial basis representation is used for the elements of the two finite field such that for $\alpha \in GF(2^{283})$

$$\begin{aligned} &\alpha = A(x) = a_{282}x^{282} + a_{281}x^{281} + \cdots + a_1x^1 + a_0 \pmod{f(x)}, \\ &\forall a_j \in GF(2). \end{aligned} \tag{5.16}$$

When $\alpha \in GF(2^{283\times4})$ then

$$\begin{aligned} &\alpha = A(x) = a_3x^3 + a_2x^2 + a_1x + a_0 \pmod{p(x)}, \\ &\forall a_j \in GF(2^{283}). \end{aligned} \tag{5.17}$$

A NIST [71] recommended irreducible polynomial such as

$$f(x) = x^{283} + x^{12} + x^7 + x^5 + 1, \tag{5.18}$$

could be used to generate the field $GF(2^{283})$ as this lead to 80 bit security level for the DLP. A better polynomial to use would be

$$f(x) = x^{283} + x^{119} + x^{97} + x^{93} + 1, \tag{5.19}$$

as an odd exponent polynomial means that the square root operation can be carried out in one clock cycle (see Section 5.5.5.1).

The polynomial for generating $GF(2^{283\times4})$ is

$$p(x) = x^4 + x + 1, \tag{5.20}$$

and it is defined over $GF(2^{283})$.

5.5.1 Addition

Addition in a binary extension field is trivial to implement in hardware. It is an array of XOR gates, one for every two bits of the operands that are to be added. Hence for $GF(2^{283})$ 283 XOR gates are required, and for $GF(2^{283\times4})$ 1132 are required.

5.5.2 Multiplication in $GF(2^{283})$

One of the main reasons that WSNs will be deployed is because the devices that make up the network will be cheap. In term of multiplication in $GF(2^{283})$ a fast bit-parallel multiplier is approximately 300,000 gates in area. This would be prohibitive in terms of manufacturing cost for a wireless sensor node. Thus, a bit-serial approach to designing the multiplier is warranted. There are two approaches to a bit-serial multiplier – an MSB–first design or LSB–first design [61].

5.5.2.1 The MSB–first multiplier

The MSB–first multiplier is based on the following observation.

$$\begin{aligned} C(x) &= A(x)B(x) \\ &= (a_{282}x^{282} + \cdots + a_1x + a_0)(b_{282}x^{282} + \cdots + b_1x + b_0) \pmod{f(x)} \\ &= b_0 + x(b_1A(x) + \cdots x(b_{280}A(x) + x(b_{281}A(x) + (xb_{282}A(x)) \pmod{f(x)}) \end{aligned} \tag{5.21}$$

This multiplier will require 283 clock cycles in order to complete – see Algorithm 7.

Algorithm 7 MSB Multiplication in $GF(2^{283})$

Require: $C(x) = A(x)B(x) \pmod{f(x)}$
  $C(x) = 0$
  $count = 0$
  while $count < 282$ **do**
    if $B(282) = 1$ **then**
      $C(x) = C(x) + A(x)$
    end if
    left shift $B(x)$
    $C(x) = xC(x) \pmod{f(x)}$
    $count = count + 1$
  end while
  if $B(282) = 1$ **then**
    $C(x) = C(x) + A(x)$
  end if
  return $C(x)$

The ASM chart for the algorithm of the MSB multiplier, mult_msb, is shown in figure A.3 and A.4 in Appendix A. The datapath circuitry is shown in Figure 5.5. Signals that are not read or driven in this diagram are inputs and outputs to the control logic, respectively. And the control logic is represented by the ASM chart and A.4 in Appendix A. This is true for all the datapath diagrams in this thesis.The datapath width is 283 bits wide and is indicated by n in the diagram, and the datapath width will always be 283 bits unless otherwise stated in all other datapath diagrams.

From Table 5.5.2.3 it can be seen that the area of the MSBmultiplier (mult_msb) is 10001 gates with a silicon area of 0.024mm^2. It has a latency of 2.32μs and consumes 3.94nJ of energy.

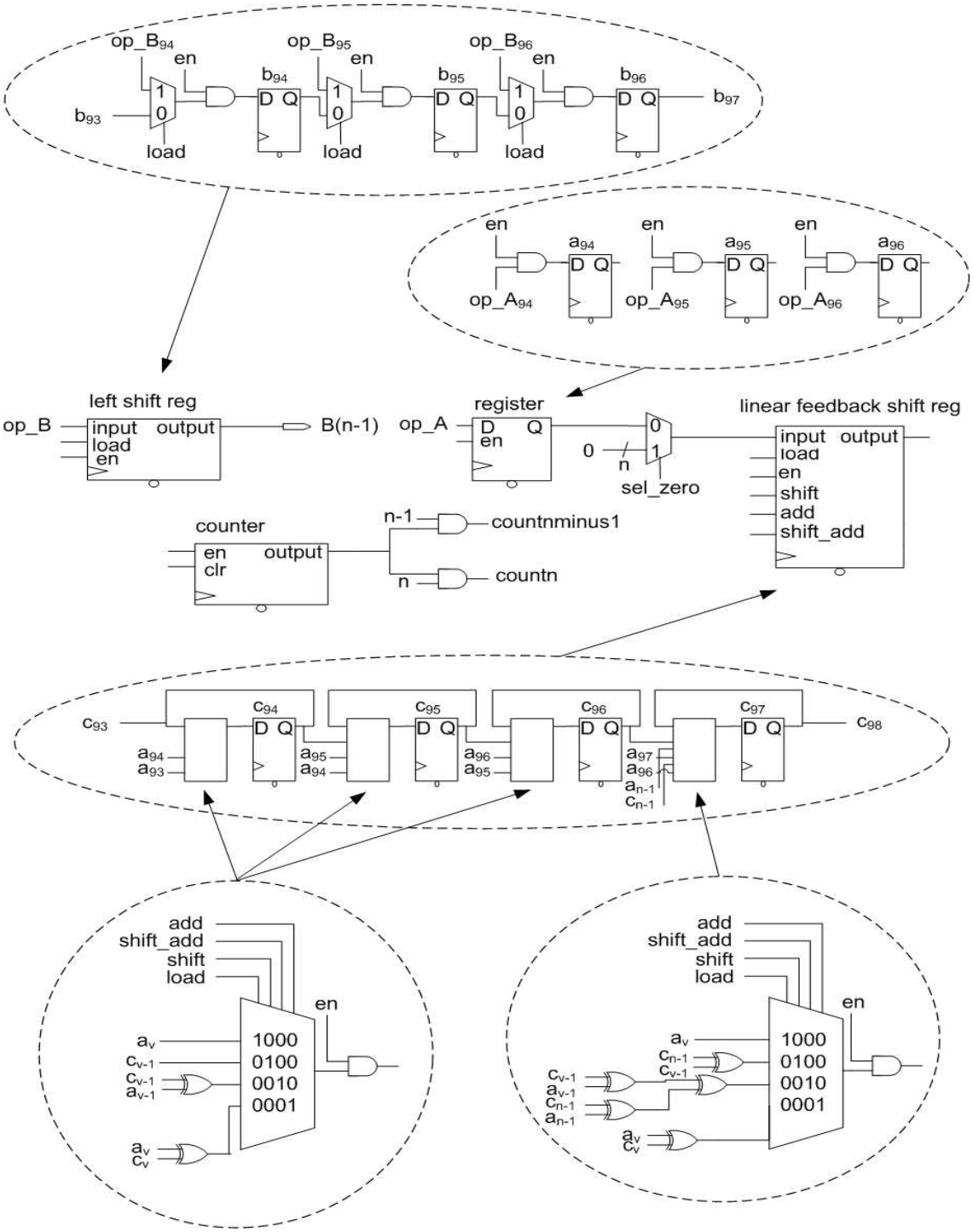

Figure 5.5: mult_msb: Datapath circuit for the MSB multiplier in $GF(2^{283})$

5.5.2.2 The LSB–first multiplier

Another approach to bit-serial multiplication in $GF(2^{283})$ is to use a LSB first multiplier. The LSB–first multiplier is based on the following observation.

$$\begin{aligned} C(x) &= A(x)B(x) \\ &= (a_{282}x^{282} + \cdots + a_1x + a_0)(b_{282}x^{282} + \cdots + b_1x + b_0) \pmod{f(x)} \\ &= (b_{282}x^{282}A(x) + \cdots + b_1xA(x) + b_0A(x)) \pmod{f(x)} \\ &= (b_{282}x^{282}A(x) \pmod{f(x)}) + \cdots \\ &+ (b_1xA(x) \pmod{f(x)}) + (b_0A(x) \pmod{f(x)}) \end{aligned} \tag{5.22}$$

$C(x)$ can be calculated using a shift and add algorithm where the first partial product is $b_0A(x)$. $B(x)$ is then shifted right one bit while at the same time $A(x)$ is multiplied by x and reduced mod $f(x)$. It is added to the previous product if b_1 is equal to one. The algorithm will terminate when the value of the right shift register is equal to zero (see Algorithm 8). This is an early exit mechanism, as it could finish after one clock cycle or 283 clock cycles. From Table 5.5.2.3 it can be seen that the area of the LSB multiplier (mult_lsb1) is 8992 gates with a silicon area of $0.022mm^2$. It has a latency of $2.31\mu s$ and consumes $4.46nJ$ of energy. Therefore, this multiplier has 11% less area and uses 13% more energy than the MSB multiplier. Using the area*energy*time metric it can be seen that the differences between the two approaches is less than 1%.

Algorithm 8 LSB Multiplication in $GF(2^{283})$

Require: $C(x) = A(x)B(x) \pmod{f(x)}$
  $C(x) = 0$
  while $B(x) \neq 0$ **do**
    $C(x) = C(x) + b_0A(x)$
    right shift $B(x)$
    $A(x) = xA(x) \pmod{f(x)}$
  end while

From an analysis of the algorithm it can seen that the addition of right shift and linear feedback barrel shift registers can be used to improve the performance of the circuit. Two, three, four or five consecutive zero bits are searched for, and the registers shifted

accordingly. As there is a cost in terms of extra area for every extra bit searched for, it was decided that five would be the most bits considered. This is because the probability of five zeros is $\frac{1}{32}$ which is quite high.

The ASM charts for the LSB multiplier algorithm and the control logic are shown in figures A.1 and A.2 in Appendix A. The datapath circuitry is shown in Figure 5.6. When the system is complete, clock tree gating will be implemented at a higher level to reduce the energy being dissipated on the clock nets.

5.5.2.3 Conclusion

From an analysis of the multipliers presented in Table 5.5.2.3, where mult_lsb2 represents the search for the two LSB being zero etc., it can be seen that mult_lsb3 has the lowest area*energy*time figure. This is the multiplier that is used in this work.

operation	area (gates)	area (mm^2)	time (μs)	time (cycles)	power (mW)	energy (nJ)	area*energy*time E-22mSJ
mult_lsb1	8992	0.022	2.31	289	1.93	4.46	2.24
mult_lsb2	9671	0.023	1.897	237	2.23	4.23	1.87
mult_lsb3	10217	0.025	1.776	222	2.33	4.14	1.8
mult_lsb4	10925	0.026	1.717	215	2.37	4.06	1.83
mult_lsb5	11713	0.028	1.708	214	2.43	4.15	2.01
mult_msb	10001	0.024	2.32	290	1.7	3.94	2.22

Table 5.1: Results for multiplication in $GF(2^{283})$

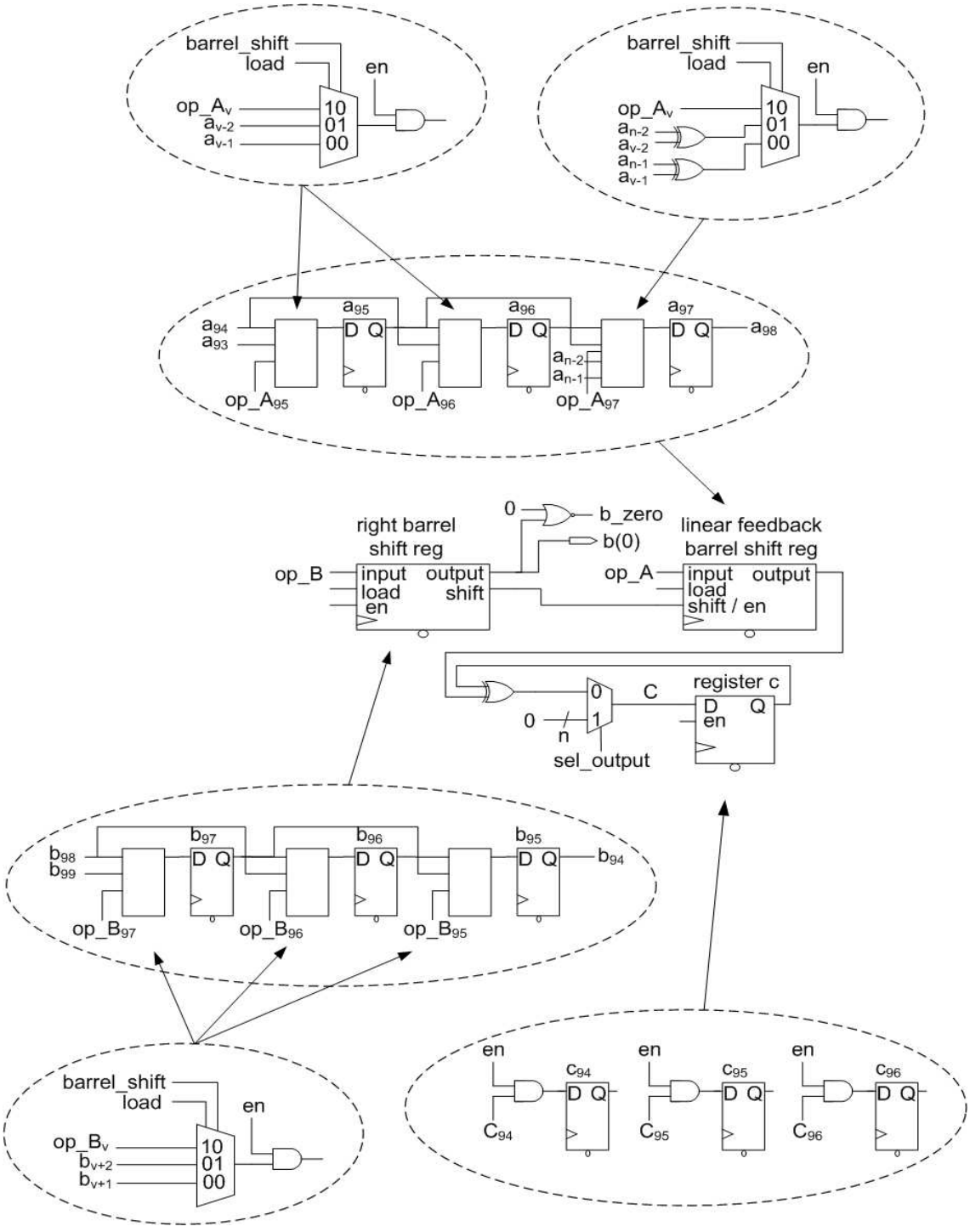

Figure 5.6: mult_lsb: Datapath circuit for the LSB multiplier in $GF(2^{283})$

5.5.3 Multiplication in $GF(2^{283\times4})$

In this section the approach taken to multiplication of two elements

$$\gamma = \alpha\beta, \quad \forall\alpha,\beta \in GF(2^{283\times4})$$

is outlined.

As the multiplication circuitry will exist for $GF(2^{283})$, it can be used to perform the multiplication for $GF(2^{283\times4})$ using Karatsuba and Ofman's algorithm [50]. This sharing of resources will lead to a decrease in the monetary cost of the system.

As the elements are represented using the polynomial basis (5.17). Then the multiplication is as follows.

$$\left(\sum_{i=0}^{3} c_i x^i\right) = \left(\sum_{i=0}^{3} a_i x^i\right)\left(\sum_{i=0}^{3} b_i x^i\right) \pmod{p(x)} \tag{5.23}$$

If normal polynomial multiplication is employed, and the terms reduced mod $p(x)$, then this would result in

$$\begin{aligned} C(x) = {} & (a_0b_0 + a_1b_3 + a_3b_1 + a_2b_2) \\ & + (a_0b_1 + a_1b_0 + a_1b_3 + a_2b_2 + a_2b_3 + a_3b_1 + a_3b_2)x \\ & + (a_0b_2 + a_1b_1 + a_2b_0 + a_2b_3 + a_3b_2 + a_3b_3)x^2 \\ & + (a_0b_3 + a_1b_2 + a_2b_1 + a_3b_0 + a_3b_3)x^3 \pmod{p(x)} \end{aligned} \tag{5.24}$$

and therefore requires sixteen multiplications and fifteen additions in $GF(2^{283})$. However, by using the Karatsuba algorithm [50, 72] the number of multiplications can be reduced. The two operands $A(x), B(x) \in GF(2^{283\times4})$ are written as,

$$\begin{aligned} A(x) &= x^2(a_3x + a_2) + (a_1x + a_0) = x^2A_h + A_l, \\ B(x) &= x^2(b_3x + b_2) + (b_1x + b_0) = x^2B_h + B_l. \end{aligned} \tag{5.25}$$

The Karatsuba algorithm partial products are,

$$\begin{aligned} D_1 &= A_l B_l, \\ D_2 &= (A_l + A_h)(B_l + B_h), \\ D_3 &= A_h B_h, \end{aligned} \tag{5.26}$$

and hence

$$\begin{aligned} C(x) = A(x)B(x) &= (x^2 A_h + A_l)(x^2 B_h + B_l), \\ &= x^4 A_h B_h + x^2 A_h B_l + x^2 A_l B_h + A_l B_l, \\ &= x^4 D_3 + x^2(D_2 - D_1 - D_3) + D_1 \pmod{p(x)}. \end{aligned} \tag{5.27}$$

The Karatsuba algorithm can be applied recursively to D_1, D_2 and D_3 such that

$$\begin{aligned} D_1 = A_l B_l &= (a_1 x + a_0)(b_1 x + b_0), \\ &= (A_{h1} x + A_{l1})(B_{h1} x + B_{l1}) \pmod{p(x)}. \end{aligned} \tag{5.28}$$

The partial products are

$$\begin{aligned} D_{11} &= A_{l1} B_{l1} = a_0 b_0, \\ D_{12} &= (A_{l1} + A_{h1})(B_{l1} + B_{h1}) = (a_0 + a_1)(b_0 + b_1), \\ D_{13} &= A_{h1} B_{h1} = a_1 b_1, \end{aligned} \tag{5.29}$$

therefore

$$\begin{aligned} D_1 &= x^2 D_{13} + x(D_{12} - D_{11} - D_{13}) + D_{11} \\ &= x^2 a_1 b_1 + x((a_0 + a_1)(b_0 + b_1) - a_0 b_0 - a_1 b_1) + a_0 b_0 \pmod{p(x)}. \end{aligned} \tag{5.30}$$

The same technique can be used for D_2

$$\begin{aligned} D_2 = (A_l + A_h)(B_l + B_h) &= ((a_1 + a_3)x + (a_0 + a_2))((b_1 + b_3)x + (b_0 + b_2)) \\ &= (A_{h2} x + A_{l2})(B_{h2} x + B_{l2}) \pmod{p(x)}. \end{aligned} \tag{5.31}$$

The partial products are

$$\begin{aligned} D_{21} &= A_{l2}B_{l2} = (a_0 + a_2)(b_0 + b_2), \\ D_{22} &= (A_{l2} + A_{h2})(B_{l2} + B_{h2}) = (a_0 + a_2 + a_1 + a_3)(b_0 + b2 + b_1 + b_3), \\ D_{23} &= A_{h2}B_{h2} = (a_1 + a_3)(b_1 + b_3), \end{aligned} \tag{5.32}$$

therefore

$$\begin{aligned} D_2 &= x^2 D_{23} + x(D_{22} - D_{21} - D_{23}) + D_{21} \\ &= x^2(a_1 + a_3)(b_1 + b_3) + x((a_0 + a_2 + a_1 + a_3)(b_0 + b_2 + b_1 + b_3) \\ &- (a_0 + a_2)(b_0 + b_2) - (a_1 + a_3)(b_1 + b_3)) \\ &+ (a_0 + a_2)(b_0 + b_2) \pmod{p(x)}. \end{aligned} \tag{5.33}$$

Finally we find for D_3

$$\begin{aligned} D_3 &= A_h B_h = (a_3 x + a_2)(b_3 x + b_2) \\ &= (A_{h3}x + A_{l3})(B_{h3}x + B_{l3}) \pmod{p(x)}. \end{aligned} \tag{5.34}$$

The Karatsuba algorithm partial products are

$$\begin{aligned} D_{31} &= A_{l3}B_{l3} = a_2 b_2, \\ D_{32} &= (A_{l3} + A_{h3})(B_{l3} + B_{h3}) = (a_2 + a_3)(b_2 + b_3), \\ D_{33} &= A_{h3}B_{h3} = a_3 b_3, \end{aligned} \tag{5.35}$$

therefore

$$\begin{aligned} D_3 &= x^2 D_{33} + x(D_{32} - D_{33} - D_{33}) + D_{33} \\ &= x^2 a_3 b_3 + x((a_2 + a_3)(b_2 + b_3) - a_2 b_2 - a_3 b_3) + a_2 b_2 \pmod{p(x)}. \end{aligned} \tag{5.36}$$

For the partial multiplication products 9 multiplications and 10 additions are required in $GF(2^{283})$.

Substituting the values for D_1 ,D_2 and D_3 into equation (5.27) and reducing the resul-

tant equation mod $p(x)$ leads to the following

$$\begin{aligned}
c_0 &= a_2b_2 + (a_1 + a_3)(b_1 + b_3) + a_0b_0 + a_3b_3 + a_1b_1, \\
c_1 &= (a_2 + a_3)(b_2 + b_3) + (a_1 + a_3)(b_1 + b_3) + (a_0 + a_1)(b_0 + b_1) + a_0b_0, \\
c_2 &= (a_2 + a_3)(b_2 + b_3) + a_1b_1 + (a_0 + a_2)(b_0 + b_2) + a_0b_0, \\
c_3 &= a_0b_0 + (a_0 + a_1)(b_0 + b_1) + a_1b_1 + (a_0 + a_2)(b_0 + b_2) \\
&\quad + (a_0 + a_2 + a_1 + a_3)(b_0 + b_2 + b_1 + b_3) \\
&\quad + (a_1 + a_3)(b_1 + b_3) + a_2b_2 + (a_2 + a_3)(b_2 + b_3).
\end{aligned} \tag{5.37}$$

Using terms common to more than one equation i.e. $a_0b_0 + a_1b_1$ it can be seen that 12 additions are required in $GF(2^{283})$. In total, 9 multiplications and 22 additions are required in $GF(2^{283})$ when the Karatsuba algorithm is employed.

The ASM charts for the multiplier algorithm and the control logic are shown in Figure A.6 in Appendix A. The datapath circuitry is shown in Figure 5.7. The datapath width is 283 bits wide. In order to reduce dynamic energy dissipation, wires are held at a constant value when not in use. This is accomplished through the signals enadd10 and enadd12 (not shown), which gate the inputs and the combinational logic, respectively. Clock tree gating is used at a higher level to reduce the energy being dissipated on the clock nets. The LSB–first multipliers clocks are to be gated with their "done" signals. This technique takes advantage of the early exit of the LSB–first multipliers due to their structure.

The multiplier has a NAND2 equivalent gate count of 123059 with an area of $0.295mm^2$. It has a latency of $1.93\mu s$, which is comparable to the LSB–first multiplier due to its parallel architecture. It also consumes $43.4nJ$ of energy.

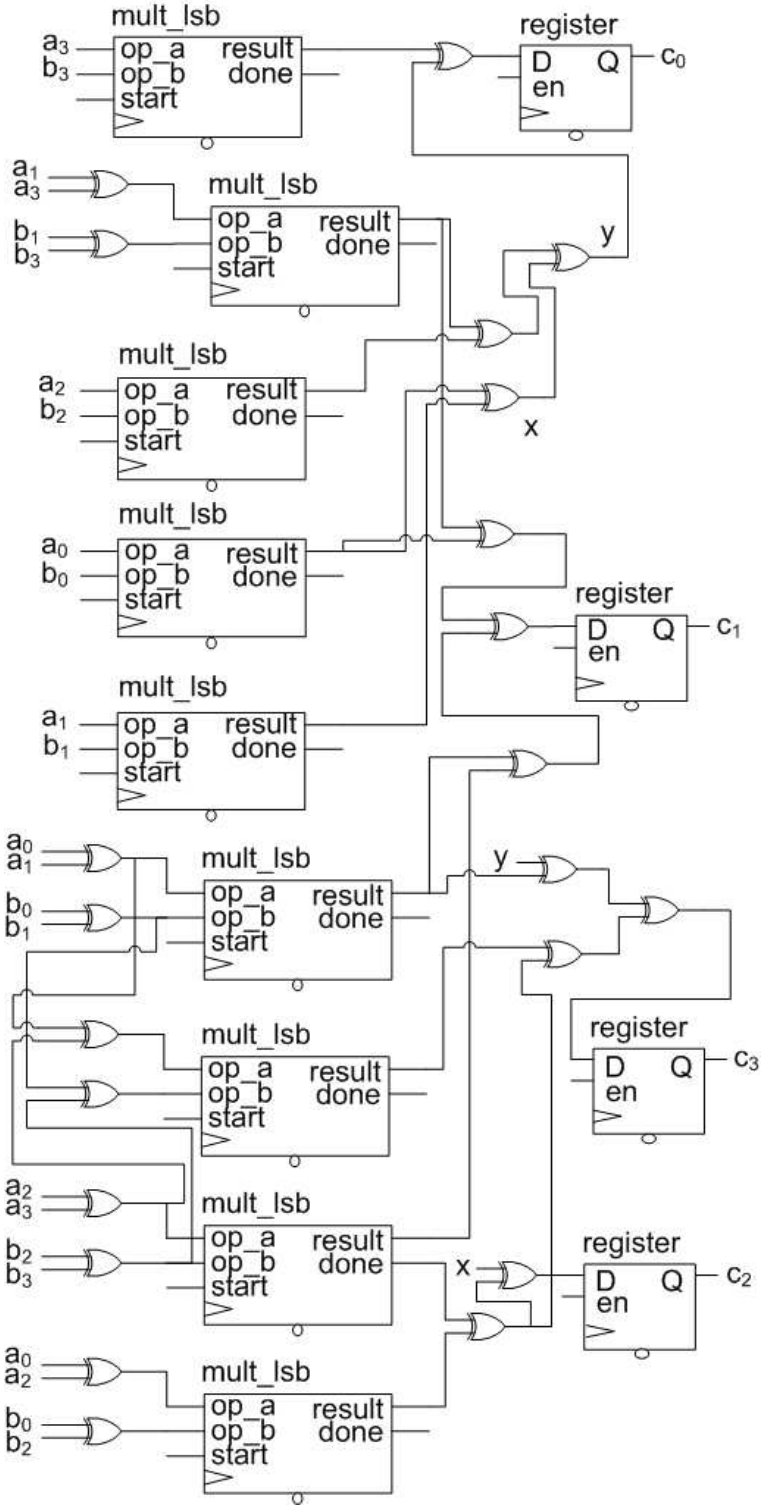

Figure 5.7: mult_koa: Datapath circuit for the multiplier in $GF(2^{283\times4})$

5.5.4 Squaring

The bit-serial multiplier described in Section 5.5.2 could be used for squaring, but as squaring is used 283 times in each loop and in the inversion circuitry, this is not the optimum choice. Instead, a bit-parallel squaring circuit has been implemented. For example, if given $C(x), A(x) \in GF(2^4)$ then

$$\begin{aligned} C(x) &= (A(x))^2 \pmod{x^4 + x + 1} \\ &= (a_3x^3 + a_2x^2 + a_1x + a_0)^2 \pmod{x^4 + x + 1} \\ &= a_3x^3 + (a_1 + a_3)x^2 + a_2x + (a_0 + a_2) \pmod{x^4 + x + 1} \end{aligned} \tag{5.38}$$

This can be represented as a binary 4×4 matrix.

$$\begin{pmatrix} c_0 \\ c_1 \\ c_2 \\ c_3 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \end{pmatrix} \tag{5.39}$$

This can be implemented with two XOR gates and a reordering of the inputs (see Figure 5.8) [61].

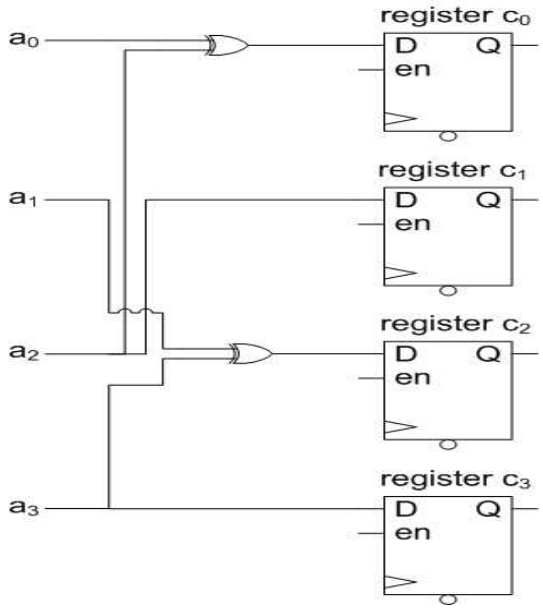

Figure 5.8: squarer: Squaring circuit for $GF(2^4)$

With the aid of a C++ program this technique can be applied to elements from $GF(2^{283})$. The resulting matrix can be converted into hardware using VHDL. As this circuit comprises only of coefficient re-ordering and additions it does not consume a lot of area – 3312 gates or $0.005mm^2$. Its latency is $0.008\mu s$ and it uses $0.004nJ$ of energy.

5.5.5 Square Root

In order to implement the Tate pairing the square root operation

$$\sqrt{\alpha} = \alpha^{2^{283-1}}, \qquad \alpha \in GF(2^{283}) \tag{5.40}$$

is required. The best approach to implementing this operation would be to calculate the inversion of the squaring binary matrix. In the case of $GF(2^{283})$ it was found that the matrix is singular and another approach had to be considered. Two approaches were considered; a bit serial approach and a bit parallel approach.

5.5.5.1 Serial square root circuit

From Equation (5.40) it can be seen that the square root can be calculated by 282 operations of the squaring circuitry, and this is what is implemented. The ASM charts for the square root circuit algorithm and the control logic are shown in Figures A.5 in Appendix A. The datapath circuitry is shown in Figure 5.9.

5.5.5.2 Parallel square root circuit

Using the techniques of Fong et al [32], it is possible to reduce the latency of the square root operation to one clock cycle. Given

$$\begin{aligned} \sqrt{\alpha} &= \alpha^{2^{m-1}} \pmod{g(x)}, \\ &= \left(\sum_{i=0}^{m-1} a_i x^i \right)^{2^{m-1}} \pmod{g(x)} \end{aligned} \tag{5.41}$$

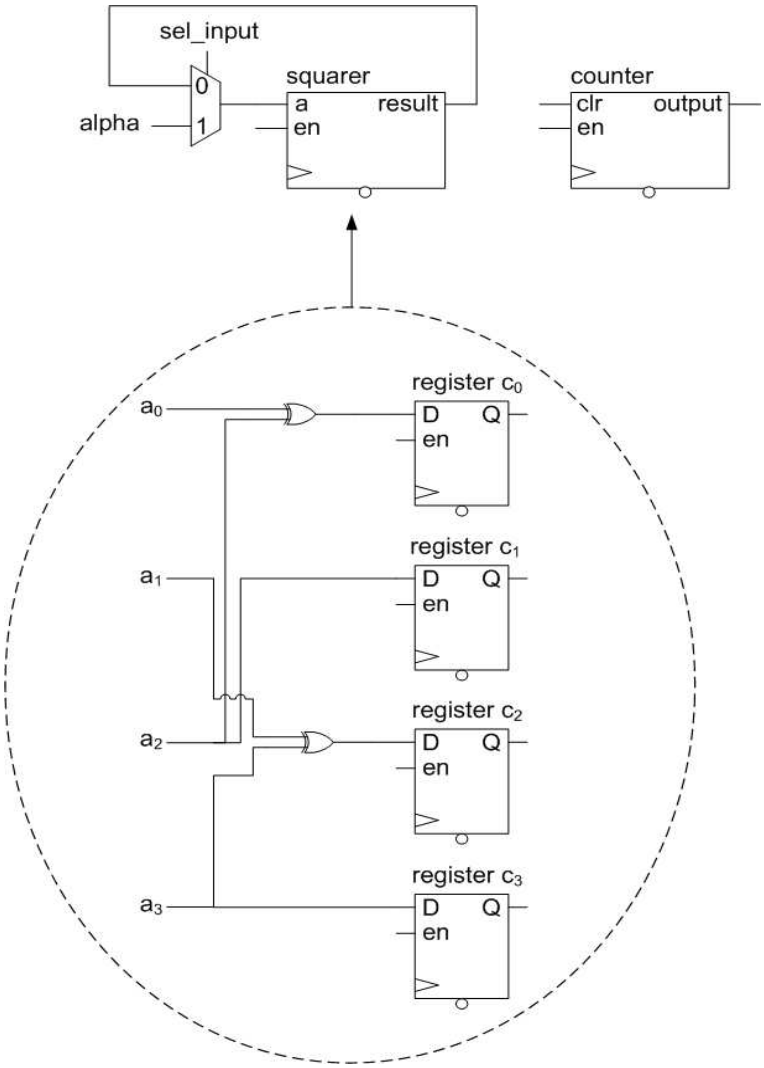

Figure 5.9: sqroot: Serial datapath circuit for the square root operation in $GF(2^{283})$

where

$$g(x) = x^m + x^t + x^u + x^v + 1, \tag{5.42}$$

and

$$\alpha = a_{m-1}x^{m-1} + a_{m-2}x^{m-2} + \cdots + a_1x^1 + a_0 \pmod{g(x)} \tag{5.43}$$

All of the exponents in Equation (5.42) are odd. Equation (5.41) can be further developed as below;

$$\begin{aligned}
\sqrt{\alpha} &= \left(\sum_{i=0}^{m-1} a_i x^i\right)^{2^{m-1}} \\
&= \sum_{i=0}^{m-1} a_i \left(x^{2^{m-1}}\right)^i \\
&= \sum_{i=0}^{(m-1)/2} a_{2i} \left(x^{2^{m-1}}\right)^{2i} + \sum_{i=0}^{(m-3)/2} a_{2i+1} \left(x^{2^{m-1}}\right)^{2i+1} \\
&= \sum_{i=0}^{(m-1)/2} a_{2i} x^i + \sum_{i=0}^{(m-3)/2} a_{2i+1} x^{2^{m-1}} x^i \\
&= \alpha_{even} + \alpha_{odd}\sqrt{x}
\end{aligned} \tag{5.44}$$

From Equation (5.42) it can be seen that

$$\begin{aligned}
1 &= x^m + x^t + x^u + x^v \pmod{g(x)} \\
x &= x^{m+1} + x^{t+1} + x^{u+1} + x^{v+1} \pmod{g(x)} \\
\sqrt{x} &= x^{(m+1)/2} + x^{(t+1)/2} + x^{(u+1)/2} + x^{(v+1)/2} \pmod{g(x)}
\end{aligned} \tag{5.45}$$

Therefore

$$\sqrt{\alpha} = \alpha_{even} + \alpha_{odd}(x^{(m+1)/2} + x^{(t+1)/2} + x^{(u+1)/2} + x^{(v+1)/2}) \tag{5.46}$$

In the case of $\alpha \in GF(2^{283})$

$$\sqrt{\alpha} = \alpha_{even} + \alpha_{odd}(x^{142} + x^{60} + x^{49} + x^{42}) \tag{5.47}$$

and the exponents are taken from Equation (5.19). This can be implemented in hardware using XOR gates in one clock cycle. The combinational logic required to do the square root operation is depicted by the ellipse with a "+" in its centre in Figure 5.10.

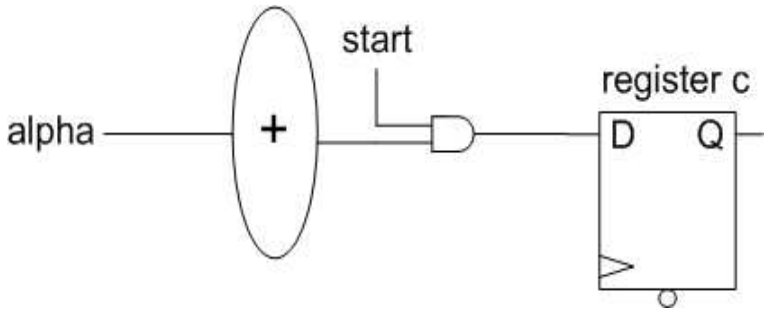

Figure 5.10: sqroot_par: Parallel square root circuit in $GF(2^{283})$

5.5.5.3 Conclusion

The area of the serial circuit is 4024 gates or 0.01mm^2. It has a latency of 2.86μs and consumes 2.12nJ. The parallel circuit preforms well when compared against the serial version. It has an area of 0.008mm^2, a latency of 0.008ms and uses 0.5nJ of energy. But most importantly its area*energy*time is five orders of magnitude lower. Therefore the parallel version will be the square root circuit that is used in this work.

operation	area (gates)	area (mm^2)	time (μs)	time (cycles)	power (mW)	energy (nJ)	area*energy*time E-22mSJ
sqroot	4024	0.01	2.863	358	0.74	2.12	0.59
sqroot_par	3218	0.008	0.008	1	0.5	0.004	2.47E-06

Table 5.2: Results for square root in $GF(2^{283})$

5.5.6 Exponentiation

The only exponentiation that is required for the Tate pairing calculation is $\beta = \alpha^{2^{283}}$ where $\alpha, \beta \in GF(2^{283\times 4})$. This is also known as the Frobenius map.

Using Equation (5.17) the exponentiation is as follows;

$$\begin{aligned} \sum_{i=0}^{3} b_i x^i &= \left(\sum_{i=0}^{3} a_i x^i \right)^{2^{283}} \\ &= \sum_{i=0}^{3} a_i^{2^{283}} x^{i2^{283}} \\ &= \sum_{i=0}^{3} a_i x^{i2^{283}} \pmod{p(x)}. \end{aligned} \tag{5.48}$$

For a proof see [62]. The proof is based on the fact that the binomial coefficients in the expansion of the above equation are a multiple of two, and therefore equal to zero in $GF(2)$. Also, $a_i^{2^{283}} \equiv a_i \pmod{f(x)}$ due to Equation (5.52).

As $x^{2^4} = x \pmod{p(x)}$, it can be shown;

$$\begin{aligned} x^{2^{283}} &= x^{2^{283 \bmod 4}} = x^{2^3} = \left(x^{2^2}\right)^2 = x^2 + 1 \pmod{p(x)} \\ \left(x^{2^{283}}\right)^2 &= (x^2+1)^2 = x^4 + 1 = x \pmod{p(x)} \\ \left(x^{2^{283}}\right)^3 &= \left(x^{2^{283}}\right)^2 \left(x^{2^{283}}\right) = (x^2+1)x = x^3 + x \pmod{p(x)} \end{aligned} \tag{5.49}$$

Using Equation (5.49) it can be seen that

$$\sum_{i=0}^{3} b_i x^i = (a_0 + a_1) + (a_2 + a_3)x + a_1 x^2 + a_3 x^3 \tag{5.50}$$

and can be represented using matrices as follows.

$$\begin{pmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \end{pmatrix} = \begin{pmatrix} 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \end{pmatrix} \tag{5.51}$$

The Frobenius map can therefore be implemented in hardware with two additions in $GF(2^{283})$ and reordering of the coefficients (see Figure 5.11.

5.5.7 Inversion in $GF(2^{283})$

There are two well-known techniques for inversion of $\beta \in GF(2^{283})$. One approach is based on Fermat's little theorem and the other uses the extended Euclidean algorithm.

5.5.7.1 Inversion by Fermat's Little Theorem

Fermat's little theorem (see Equation (5.52)) can be used to invert an element of $GF(2^{283})$

$$\beta^{2^{283}-1} \equiv 1 \pmod{f(x)}. \tag{5.52}$$

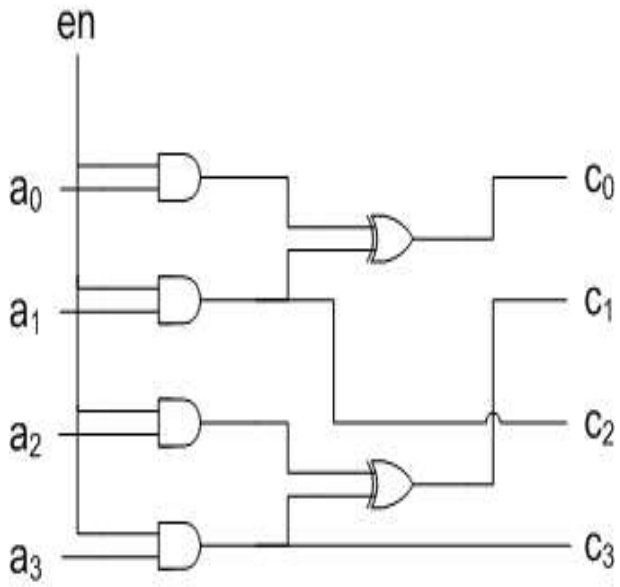

Figure 5.11: frob: Exponentiation circuit in $GF(2^{283\times 4})$

This means that $\beta^{2^{283}-2}\beta \equiv 1 \pmod{p(x)}$ and therefore $\beta^{2^{283}-2}$ is the inverse of β. The inverse of β can be calculated with the square and multiply technique using the following observations.

$$\begin{aligned}\beta^{-1} = \beta^{2^{283}-2} &= \beta^{2^1}\beta^{2^2}\beta^{2^3}\cdots\beta^{2^{282}} \\ &= (\cdots(((\beta)^2\beta)^2\beta)^2\cdots\beta)^2\end{aligned} \tag{5.53}$$

This algorithm requires 282 squarings and 281 multiplications in $GF(2^{283})$. From section 5.5.2 and 5.5.4, it can be seen that multiplication in $GF(2^{283})$ is a very expensive operation in terms of time and energy. It would be beneficial to reduce the number of multiplications. This can achieved using the techniques of Itoh and Tsujii [49].

As

$$\beta^{-1} = \beta^{2^{283}-2} = \left(\beta^{2^{283-1}-1}\right)^2$$

or more generally

$$\beta^{-1} = \beta^{2^n-2} = \left(\beta^{2^{n-1}-1}\right)^2$$

for a field $GF(2^n)$ then we can apply the following recursive formula to reduce the num-

ber of multiplications. When n is odd then

$$\beta^{2^{n-1}-1} = \left(\beta^{2^{\frac{n-1}{2}}-1}\right)^{2^{\frac{n-1}{2}}} \beta^{2^{\frac{n-1}{2}}-1} \tag{5.54}$$

and when n is even

$$\beta^{2^{n-1}-1} = \left(\beta^{2^{n-2}-1}\right)^{2} \beta \tag{5.55}$$

Equation (5.54) can be proved as follows;

$$\begin{aligned}
\left(\beta^{2^{\frac{n-1}{2}}-1}\right)^{2^{\frac{n-1}{2}}} \beta^{2^{\frac{n-1}{2}}-1} &= \beta^{2^{\frac{n-1}{2}}\,2^{\frac{n-1}{2}} - 2^{\frac{n-1}{2}}} \beta^{2^{\frac{n-1}{2}}-1} \\
&= \beta^{2^{\frac{n-1}{2}+\frac{n-1}{2}} - 2^{\frac{n-1}{2}}} \beta^{2^{\frac{n-1}{2}}-1} \\
&= \beta^{2^{n-1} - 2^{\frac{n-1}{2}}} \beta^{2^{\frac{n-1}{2}}-1} \\
&= \beta^{2^{n-1} - 2^{\frac{n-1}{2}} + 2^{\frac{n-1}{2}}-1} \\
&= \beta^{2^{n-1}-1}
\end{aligned}$$

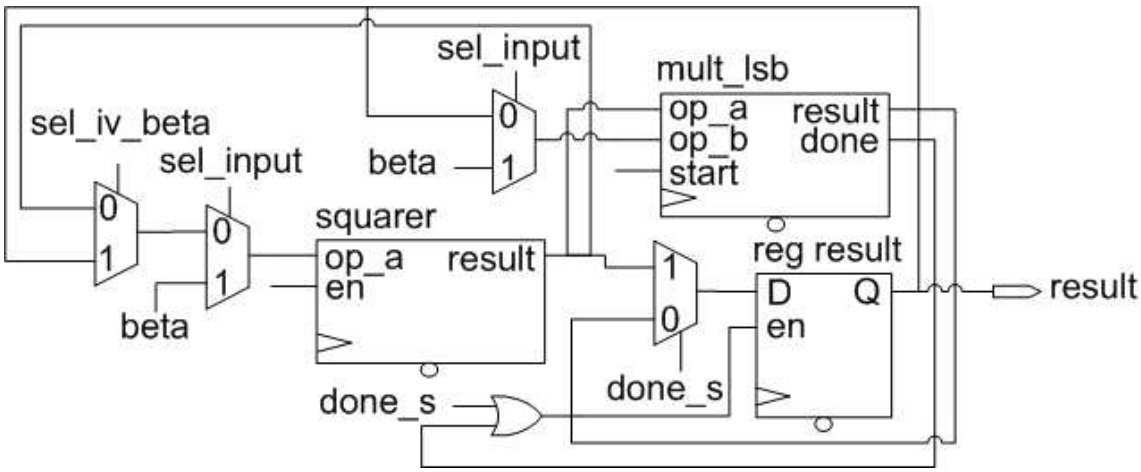

Figure 5.12: Datapath circuit for the inverter in $GF(2^{283})$using Fermat's little theorem

The same approach can be used to prove Equation (5.55) .

β^{-1} can now be decomposed using Equations (5.54) and (5.55)

$$\begin{aligned}
\beta^{2^{283}-2} &= \left(\beta^{2^{283-1}-1}\right)^2 \\
\beta^{2^{283-1}-1} &= \left(\beta^{2^{141}-1}\right)\left(\beta^{2^{141}-1}\right)^{2^{141}} \\
\beta^{2^{142-1}-1} &= \beta\left(\beta^{2^{140}-1}\right)^2 \\
\beta^{2^{141-1}-1} &= \left(\beta^{2^{70}-1}\right)\left(\beta^{2^{70}-1}\right)^{2^{70}} \\
\beta^{2^{71-1}-1} &= \left(\beta^{2^{35}-1}\right)\left(\beta^{2^{35}-1}\right)^{2^{35}} \\
\beta^{2^{36-1}-1} &= \beta\left(\beta^{2^{34}-1}\right)^2 \\
\beta^{2^{35-1}-1} &= \left(\beta^{2^{17}-1}\right)\left(\beta^{2^{17}-1}\right)^{2^{17}} \\
\beta^{2^{18-1}-1} &= \beta\left(\beta^{2^{16}-1}\right)^2 \\
\beta^{2^{17-1}-1} &= \left(\beta^{2^{8}-1}\right)\left(\beta^{2^{8}-1}\right)^{2^{8}} \\
\beta^{2^{9-1}-1} &= \left(\beta^{2^{4}-1}\right)\left(\beta^{2^{4}-1}\right)^{2^{4}} \\
\beta^{2^{5-1}-1} &= \left(\beta^{2^{2}-1}\right)\left(\beta^{2^{2}-1}\right)^{2^{2}} \\
\beta^{2^{3-1}-1} &= \beta\beta^2
\end{aligned} \tag{5.56}$$

Therefore by Equation (5.56) only 11 multiplications and 282 squarings are required to obtain the inverse of β.

The ASM charts for the inverter and the control logic are shown in figures A.7 and A.8 in Appendix A. The datapath circuitry is shown in Figure 5.12.

5.5.7.2 Inversion by the Extended Euclidean Algorithm

It is well known that the Euclidean algorithm can be used to find the greatest common denominator $\gcd$ of two integers. For example the $\gcd(87, 53)$ it equal to $\gcd(53, 34)$ i.e. the greatest common denominator of the quotient and remainder when 87 is divided by 53. This process can be applied recursively to arrive at the $\gcd$, which can be seen

from Equation (5.57) to be 1.

$$\begin{aligned} 87 &= 1 \times 53 + 34 \\ 53 &= 1 \times 34 + 19 \\ 34 &= 1 \times 19 + 15 \\ 19 &= 1 \times 15 + 4 \\ 15 &= 3 \times 4 + 3 \\ 4 &= 1 \times 3 + 1 \\ 3 &= 3 \times 1 \end{aligned} \tag{5.57}$$

Working backwards from $1 = 4 - 1 \times 3$ and using Equation (5.58), leads to $1 = 23 \times 53 - 14 \times 87$. This can means that $14 \times 87 \equiv 1 \pmod{53}$, in other words, 14 is the inverse of 87 modulo 53.

$$\begin{aligned} 1 &= 4 \times (53 - 3 \times 4) \\ &= 4 \times 4 - 15 \\ &= 4 \times (19 - 15) - 15 \\ &= 4 \times 19 - 5 \times 15 \\ &= 4 \times 19 - 5 \times (34 - 19) \\ &= 9 \times 19 - 5 \times 34 \\ &= 9 \times (53 - 34) - 5 \times 34 \\ &= 9 \times 53 - 14 \times 34 \\ &= 9 \times 53 - 14 \times (87 - 53) \\ &= 23 \times 53 - 14 \times 87 \end{aligned} \tag{5.58}$$

Using the same approach an inverse of an element of a Galois field can be obtained. Given an irreducible polynomial $f(x) = x^4 + x + 1 \in F_2[x]$ and another element $A(x) = x^3 + x \in F_2[x]$, then $\gcd(A(x), f(x))$ is equal to $\gcd(A(x), R(x))$, where $R(x)$ is the remainder of $f(x)$ divided by $A(x)$. As above, this leads to the $\gcd$ as shown by Equation

(5.59), where the $\gcd$ is the 1 as expected.

$$\begin{aligned} x^4 + x + 1 &= x(x^3 + x) + (x^2 + x + 1) \\ x^3 + x &= x(x^2 + x + 1) + x^2 \\ x^2 + x + 1 &= 1.x^2 + (x + 1) \\ x^2 &= x(x + 1) + x \\ x + 1 &= 1.x + 1 \\ x &= 1.x \end{aligned} \tag{5.59}$$

Instead of working backwards to calculate the inverse $A(x)^{-1}$ it is possible to use the Extended Euclidean Algorithm (EEA) to represent the remainder of each stage of the calculation in terms of $A(x)$ and $f(x)$, thus leading to the inverse, which in this case is $x^3 + x^2$. This is because the last line in Equation (5.60)) can be rewritten as $(x^3 + x)(x^3 + x^2) \equiv 1 \pmod{x^4 + x + 1}$.

$$\begin{aligned} x^2 + x + 1 &= (x^3 + x)x + (x^4 + x + 1) \\ x^2 &= (x^3 + x)(x + x^2) + (x^4 + x + 1)x \\ x + 1 &= (x^3 + x)(x^2 + x + 1) + (x^4 + x + 1)(x + 1) \\ x &= (x^3 + x)(x^3 + x + 1) + (x^4 + x + 1)x^2 \\ 1 &= (x^3 + x)(x^3 + x^2) + (x^4 + x + 1)(x^2 + x + 1) \end{aligned} \tag{5.60}$$

The ASM charts for the inverter and the control logic are shown in figures A.9 and A.10 in Appendix A. The datapath circuitry is shown in Figure 5.13. This block uses the degree sub-block to measure the degree of the polynomials u and v, and this circuit is also described in Appendix A.

5.5.7.3 Conclusion

The inverter based on Fermat's little theorem (inv_ferm) is 18412 NAND2 gates and $0.044mm^2$. It uses $68.1nJ$ of energy and require $22.65\mu s$ to run to completion. From Table 5.5.7.3 it can be seen that the area of Euclidean inverter (euclid) is 43% more. It

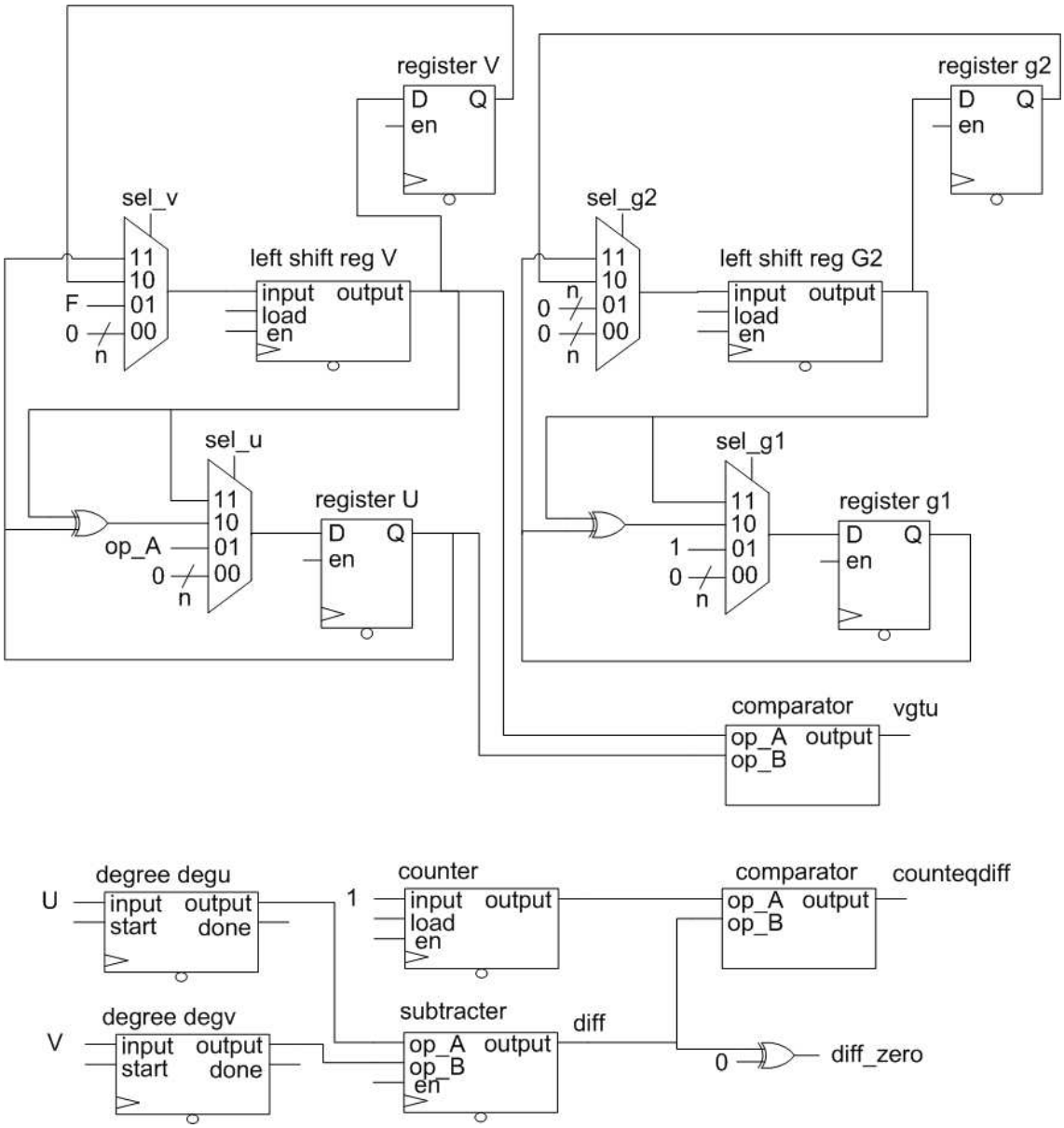

Figure 5.13: Datapath circuit for the inverter in $GF(2^{283})$using the Extended Euclidean Algorithm

is also an order of magnitude slower and uses two magnitudes more of energy. Thus, by all the metrics used to evaluate designs in this thesis, the inverter based on Fermat's little theorem is the preferred choice for inversion in $GF(2^{283})$.

5.5.8 Inversion in $GF(2^{283\times4})$

Fermat's little theorem (see Equation (5.52)) can also be used to get the inversion of an element $\alpha \in GF(2^{283\times4})$ where the extension field of $GF(2^{283})$ is obtained using the irreducible polynomial given in Equation (5.20). The technique below, that has been used

Algorithm 9 Extended Euclidean Algorithm for inversion in $GF(2^{283})$[39]

Require: $C(x) = A(x)^{-1} \pmod{f(x)}$
  $u = A(x), v = f(x), g_1 = 1, g_2 = 0$
  while $u \neq 1$ **do**
    $j = 0$
    if $v > u$ **then**
      $u = v, v = u, g_1 = g_2, g_2 = g_1$
    else
      j is equal to degree of u minus degree of v
    end if
    $u = u + x^j v$
    $g_1 = g_1 + x^j g_2$
  end while
  return $C(x) = g_1$

operation	area (gates)	area (mm^2)	time (μs)	time (cycles)	power (mW)	energy (nJ)	area*energy*time E-22mSJ
inv_ferm	18412	0.044	22.65	2831	3.01	68.1	683
euclid	26300	0.063	341.436	42680	3.02	1030	2.23E+05

Table 5.3: Results for inversion in $GF(2^{283})$

by Paar and Guajardo [37], is used as it makes use of circuits that are already designed.

If the general case $\alpha \in GF(2^{mk})$ is considered, then the inverse is

$$\alpha^{-1} = \alpha^{2^{mk}-2} = \alpha^{\frac{2^{mk}-1}{2^m-1}(2^m-1)-1} = \alpha^{r(2^m-2)+r-1} = (\alpha^r)^{-1}\alpha^{r-1}$$

where $r = \frac{2^{mk}-2}{2^m-1}$. The technique is based on the fact that

$$\alpha^r \in GF(2^m), \qquad \forall \alpha \in GF(2^{mk}) \tag{5.61}$$

Algorithm 10 Inversion in $GF(2^{mk})$

Require: $\beta = \alpha^{-1}$
  α^{r-1}
  $\alpha^r = \alpha^{r-1}\alpha$
  $(\alpha^r)^{-1}$
  $(\alpha^r)^{-1}\alpha^{r-1}$

When working in the field $GF(2^{283\times 4})$ the first stage of the inversion algorithm (10)

is obtained by the following equations.

$$r - 1 = 2^{283} + (2^{283})^2 + (2^{283})^3.$$

If we let

$$\beta = \alpha^{r-1} = \alpha^{2^{283}+(2^{283})^2+(2^{283})^3}$$

where $\alpha, \beta \in GF(2^{283\times4})$ this can be rewritten as

$$\beta = \left(\left(\alpha^{2^{283}}\alpha\right)^{2^{283}}\alpha\right)^{2^{283}}$$

Three 2^{283} exponentiations (see section 5.5.6) and two multiplications in $GF(2^{283\times4})$ (see section 5.5.3) are required to perform this operation. The next stage is multiplication in $GF(2^{283\times4})$. As $\alpha \in GF(2^{283})$, $(\alpha^r)^{-1}$ is inversion in $GF(2^{283})$ as outlined earlier in this section, and finally the last step is multiplication in $GF(2^{283\times4})$.

The ASM charts for the inverter and the control logic are shown in figures A.13 and A.14 in Appendix A. The datapath circuitry is shown in Figure 5.14. This is a large circuit

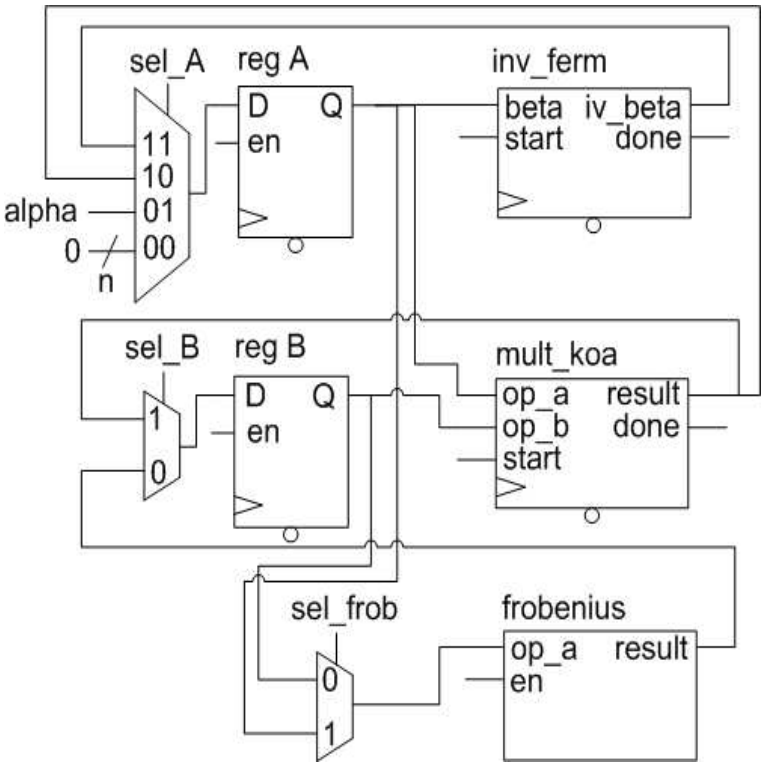

Figure 5.14: Datapath circuit for the inverter in $GF(2^{283\times4})$

with a gate count of 174048 and an area of 0.417mm^2. It has a latency of 29.8μs and its energy usage is 5.41nJ.

5.5.9 Conclusion

All of the circuits discussed in this section are listed in Table 5.5.9 where; mult_lsb* is the LSB–first $GF(2^{283})$ multiplier, mult_msb is the MSB–first $GF(2^{283})$ multiplier, square is the squaring circuit, sqroot is the square root operation, mult_koa is the Karatsuba algorithm multiplier, inv_ferm is the $GF(2^{283})$ inverter and inv_gf2m4 is the $GF(2^{283\times4})$ multiplier. The $GF(2^{283})$ multiplier is the building block upon which all the other arithmetic operations are based, along with the squaring circuit. This approach was taken as design re-use is key to reducing the cost of the overall system in terms of its area and engineering man-hours.

operation	area (gates)	area (mm^2)	time (μs)	time (cycles)	power (mW)	energy (nJ)	area*energy*time E-22mSJ
mult_lsb1	8992	0.022	2.31	289	1.93	4.46	2.24
mult_lsb2	9671	0.023	1.897	237	2.23	4.23	1.87
mult_lsb3	10217	0.025	1.776	222	2.33	4.14	1.8
mult_lsb4	10925	0.026	1.717	215	2.37	4.06	1.83
mult_lsb5	11713	0.028	1.708	214	2.43	4.15	2.01
mult_msb	10001	0.024	2.32	290	1.7	3.94	2.22
mult_koa	123059	0.295	1.926	241	22.55	43.4	247
inv_ferm	18412	0.044	22.65	2831	3.01	68.1	683
euclid	26300	0.063	341.436	42680	3.02	1030	2.23E+05
inv_gf2m4	174048	0.417	29.863	3733	18.12	541	67500
square	3312	0.005	0.008	1	0.53	0.004	1.60E-6
sqroot	4024	0.01	2.863	358	0.74	2.12	0.59
sqroot_par	3218	0.008	0.008	1	0.5	0.004	2.47E-06

Table 5.4: Results for $GF(2^{283})$ and $GF(2^{283\times4})$ arithmetic primitives

5.6 Summary of Contributions

Using a TSMC 65nm low power process various arithmetic operations in the fields $GF(2^{283})$ and $GF(2^{283\times4})$ are designed, implemented and evaluated in terms of their area, energy and latency. These are the arithmetic operations that are required for the implementation of the Tate pairing calculation.

The LSB and MSB bit serial approach to multiplication in the field $GF(2^{283})$ are evaluated and the addition of a barrel shifter to the LSB multiplier is shown to improve its performance. A parallel implementation of the $GF(2^{283})$ multiplier was investigated and found to perform well when evaluated against the area*energy*time metric. A module reuse strategy is employed to reduce the area of the design and the design effort – both important factors in the overall design's cost. Therefore, the bit serial $GF(2^{283})$ multiplier was used for the $GF(2^{283\times4})$ multiplication, which uses the Karatsuba algorithm.

Inversion in $GF(2^{283})$ is performed using the techniques of Itoh and Tsujii. It has been shown that using the metric used in this thesis that it outperforms inversion circuits based on the EEA. In the larger field, inversion is also based on Fermat's little theorem so as to make use of circuits that are already designed. Squaring and square root circuits are also designed and evaluated.

CHAPTER 6

Tate Pairing Accelerator

6.1 Introduction

In this chapter the arithmetic units presented previously are incorporated into a Tate pairing co-processor. Architectural variations are also presented and the work is evaluated

6.2 Tate Architecture

A top-level block diagram of the architecture of our proposed accelerator is given in Figure 6.1. The device is connected with the host using the Advanced Peripheral Bus (APB) protocol. The Host Interface block interfaces with the APB. It is responsible for decoding the write data input signal and address signal to write to the internal registers. The read data output is muxed in this block with the control, status and resultant data from the Tate pairing accelerator. These internal buses are gated when not in use to conserve energy.

The control and status register block is used to implement the control register, status register, and the read and write data pointers (see Appendix B). Accessing the data registers in this block has the effect of updating the write or read pointers, which are implemented as six bit counters. Writes and reads to the data registers are in fact writes and read to registers in the Tate block.

There are two clock domains in the design; the APB clock which is PCLK and runs at

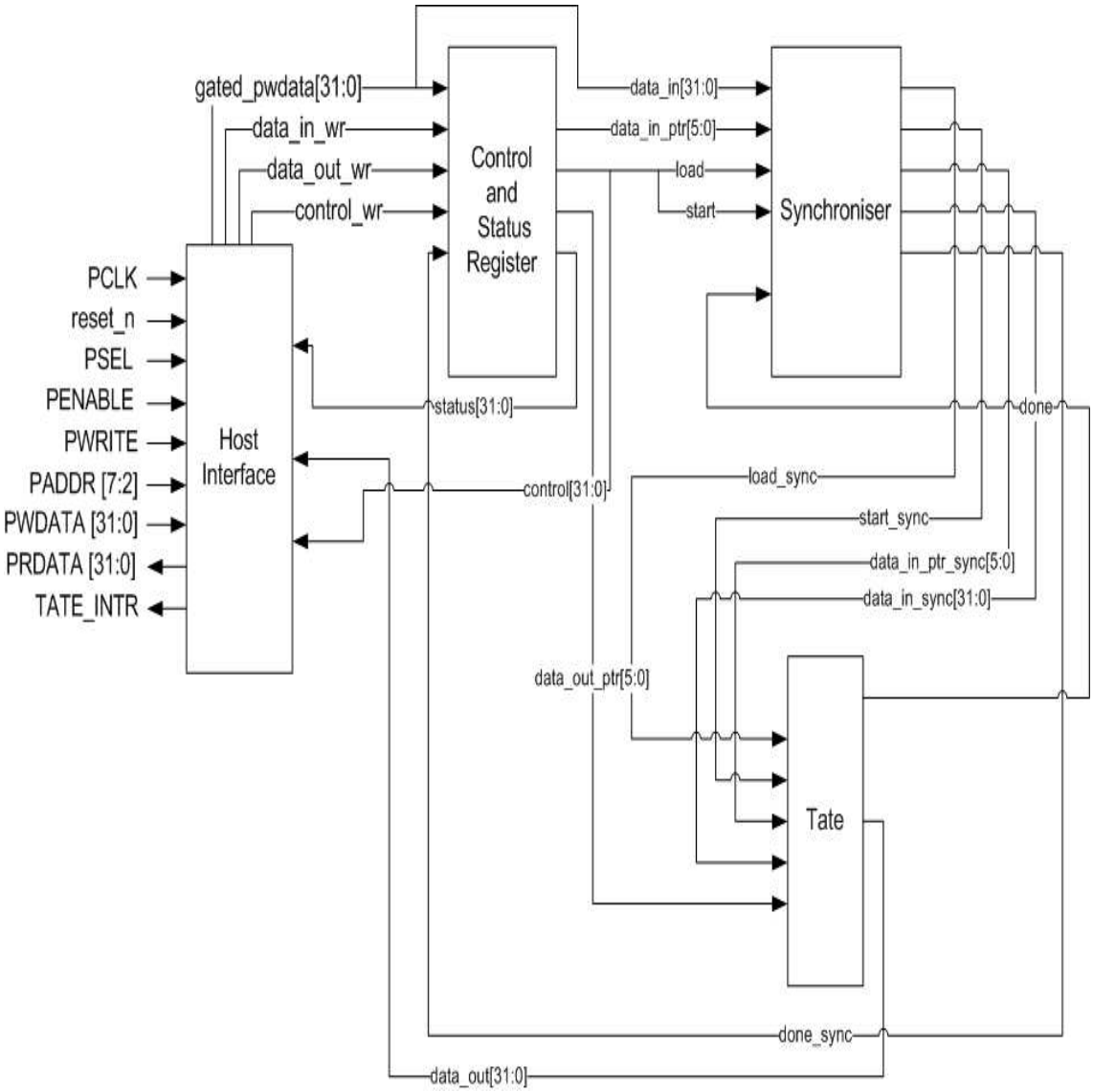

Figure 6.1: Tate Pairing Hardware Accelerator Architecture

50MHz and a system clock, tate_clk, which runs at 200MHz. These clocks are assumed to be asynchronous. Signals that cross the clock boundary will require synchronisation and this is achieved in the synchronisation block.

Finally, the Tate block implements the algorithm for calculating the Tate pairing as discussed below. Two points on the curve are required to calculate the Tate pairing and these will have coordinates in $GF(2^{283})$. Therefore four 283 bit values have to be written into the device before the Tate block can be initiated. When it is finished it sends an interrupt back to the host interface (tate_intr).

6.2.1 Algorithmic analysis

From an analysis of Algorithm 6 and the results of the arithmetic primitives presented in Table 5.5.9, an architecture for the Tate pairing accelerator is arrived at. The ASM for this architecture is shown in Figure 6.2. After the main body of the loop, $C(x) = C(x)^{2^{2\times 283}-1}$ is calculated, and this can be re-written as $C(x) = (C(x)^{2^{283}})^{2^{283}})C(x)^{-1}$. Therefore, it is calculated using two exponentiations (shown as frob in ASM) and one inversion in $GF(2^{283\times 4})$.

6.2.2 Datapath

The datapath is presented in Figure 6.3. The databus is 283×4 bits wide. The tate_in register is arranged in 5×283 bit registers that can be written to in 32bit or 283bit mode. When the elliptic curve points are written into the accelerator they are accessed in 32bit mode by the processor via the host interface block. When the operation is finished a "done" signal is asserted which is assigned to an interrupt. Upon detection of this interrupt the processor can read the output data, 32bits at a time, from the tate_out register via reads to the tate_data_out register.

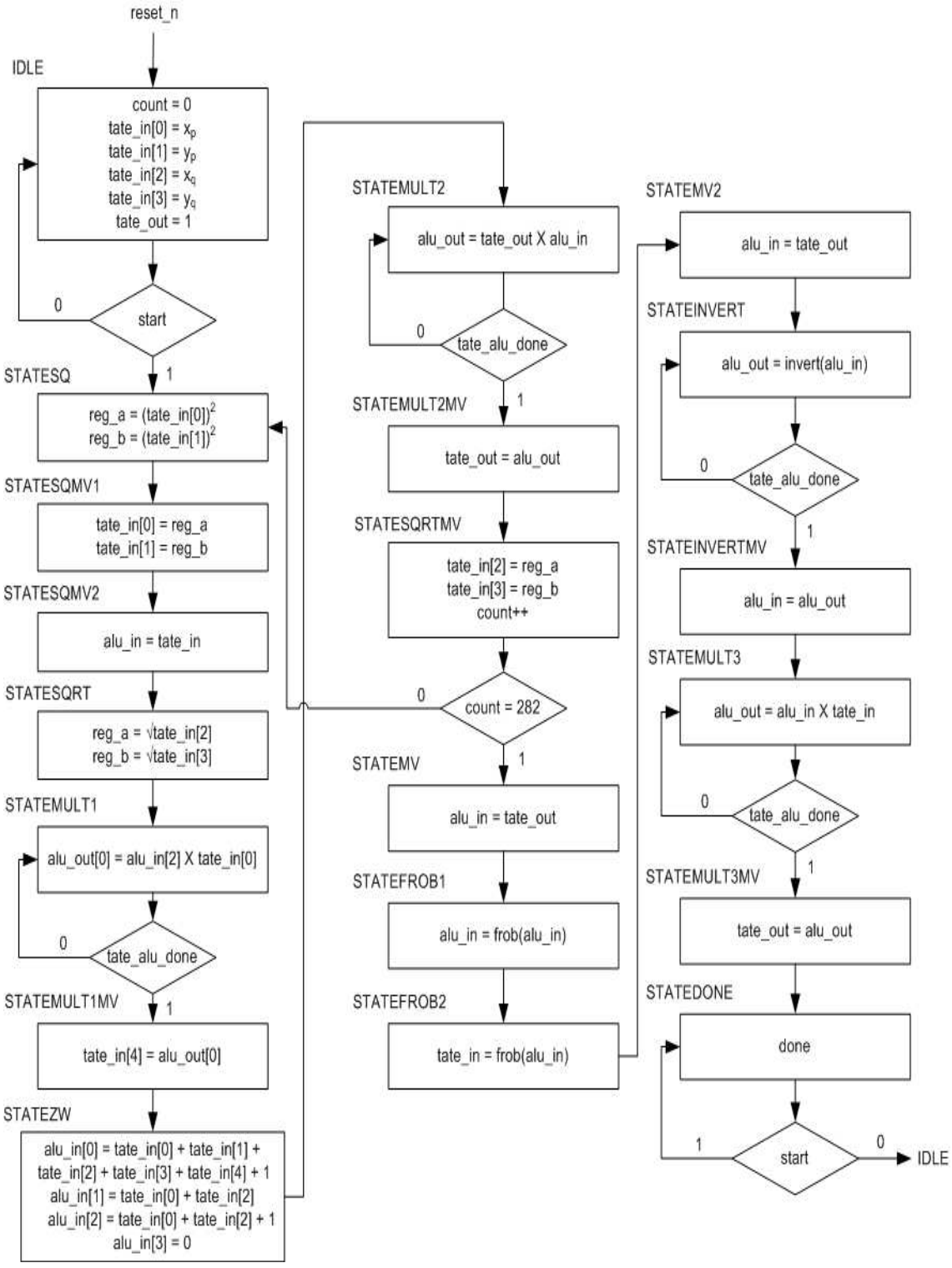

Figure 6.2: ASM for the Tate Pairing Hardware Accelerator

The zw_alu Arithmetic logic Unit (ALU) calculates $z = x_p + x_q$ and $w = z + x_p x_q + y_p + y_q + 1$. Internally, it has two squaring and square root circuits instantiated. It first is used to calculate $x_p = x_p^2$ and $y_p = y_p^2$. The tate_alu ALU then calculates $x_p x_q$. Then z and w can be arrived at. As well as the above operations, this ALU also calculates $x_q = x_q^{2^{283-1}}$ and $y_q = y_q^{2^{283-1}}$.

The tate_alu ALU calculates the multiplication in $GF(2^{283})$ and is also responsible for all other arithmetic operations required by the algorithm such as; multiplication, inversion and exponentiation – all in $GF(2^{283\times4})$. As was discussed above, the inversion circuit in $GF(2^{283\times4})$ requires an inverter in $GF(2^{283})$ and this is implemented in this ALU.

There are two sub-circuits implemented in the tate_alu; one for calculating exponentiation and the other for multiplication in both fields and inversion in $GF(2^{283})$. This last circuit is based upon the Karatsuba algorithm multiplier but with the modification that one of the $GF(2^{283})$ multipliers is replaced by a module that performs inversion and multiplication in $GF(2^{283})$.

6.2.3 Control Logic

The ASM for the control logic is shown in Figure 6.4. It is implemented as a Moore type finite state machine (FSM) with nineteen states. The data is written into the Tate block by write to the tate_load bit in the control register, whilst at the same time data is written to the tate_data_in register.

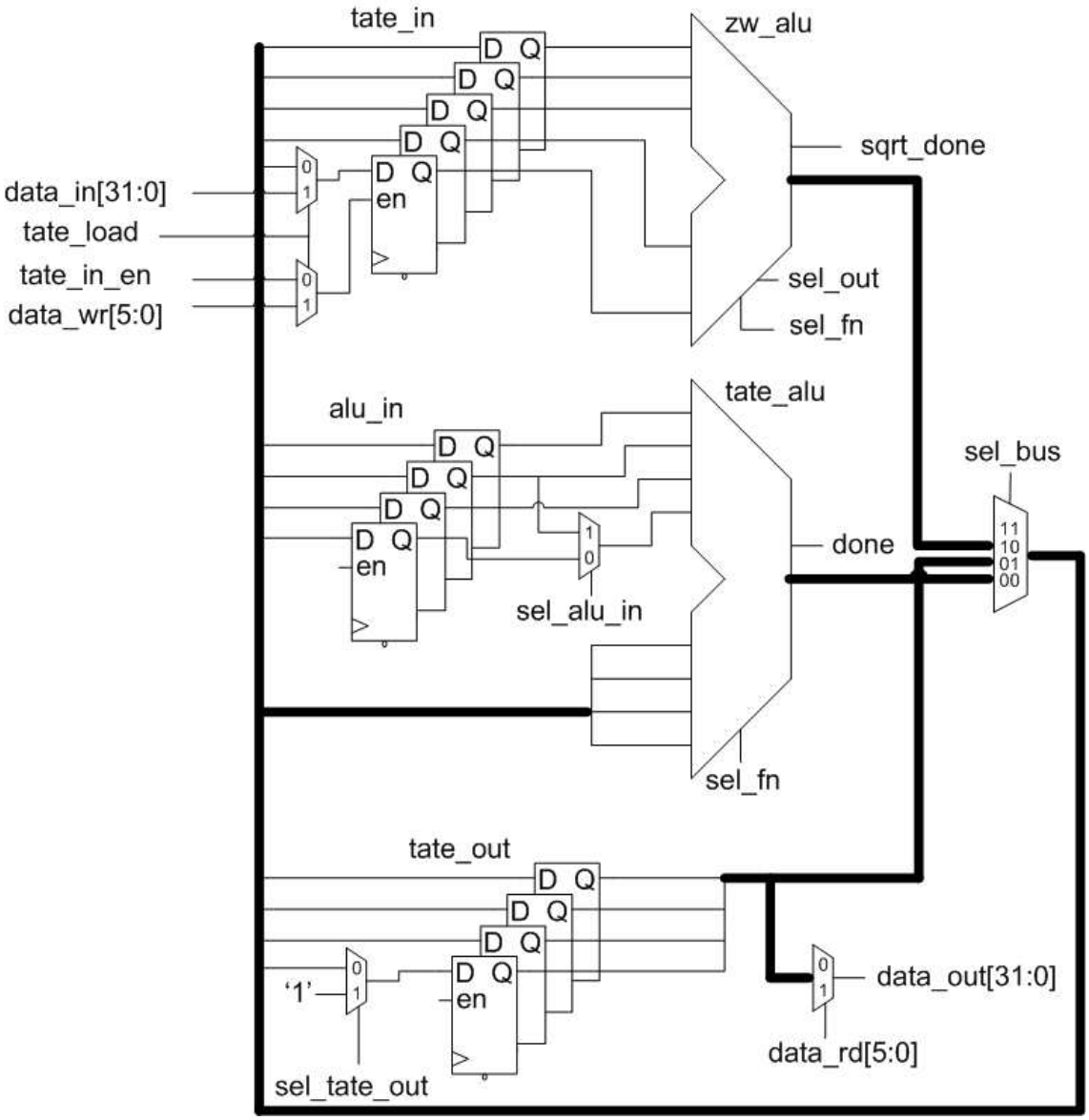

Figure 6.3: Datapath for the Tate Pairing Hardware Accelerator

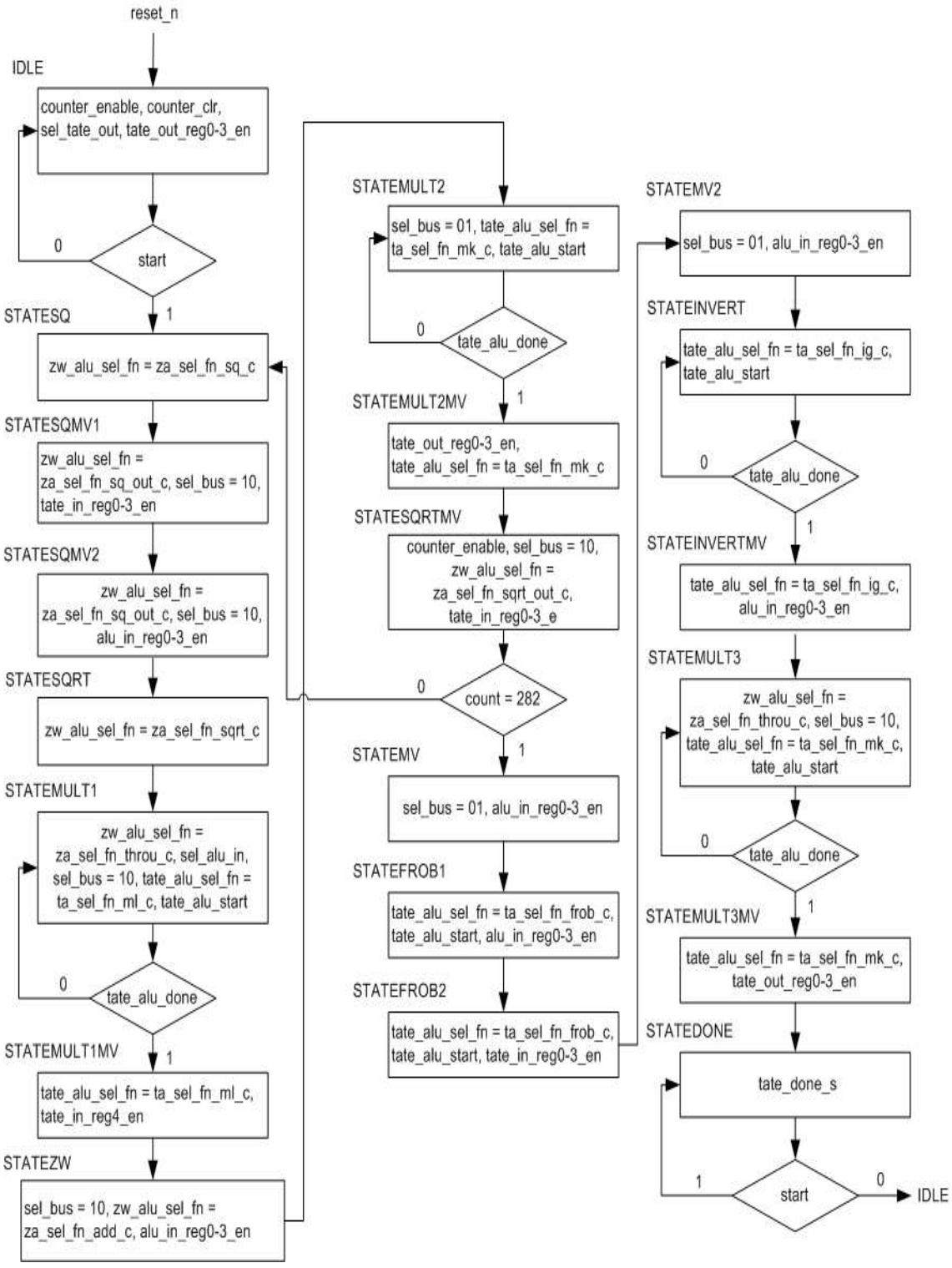

Figure 6.4: ASM for the control circuit for the Tate Pairing Hardware Accelerator

A write to the tate_start bit in the control register initiates the FSM. It does 283 loops of the main body of the algorithm and this is controlled by a counter. When the FSM is finished the tate_done signal is asserted and this is recorded in the status register and sent to the processor as an interrupt.

6.2.4 Experimental Results

Results for the Tate pairing accelerator are shown in Table 6.2.4. From this it can be seen that it has a latency of 0.7 ms, an area of 0.574mm^2 and consumes 29600nJ.

operation	area (gates)	area (mm^2)	time (μs)	time (cycles)	power (mW)	energy (nJ)	area*energy*time E-22mSJ
tate	250992	0.574	698.11	139622	42.4	29600	1.19E+08

Table 6.1: Results for the Tate pairing accelerator

6.3 Architectural Variations

There is a lot of memory that is used in this design. Due to the limitations of the design library (it did not include RAM cells) this memory was implemented as D-type registers. A better approach would be to use local SRAM as this would reduce the overall area and therefore the energy used by the accelerator.

Even though a bit-parallel multiplier was discounted at an early stage in the research due to its large area, it has been found that it performs well when evaluated using the area*energy*time metric against a bit-serial approach. Given $C(x), A(x), B(x) \in GF(2^4)$

then $C(x)$ can be written as in Equation (6.1).

$$\begin{aligned} C(x) &= A(x)B(x) \pmod{x^4+x+1} \\ &= (a_3x^3+a_2x^2+a_1x+a_0)(b_3x^3+b_2x^2+b_1x+b_0) \pmod{x^4+x+1} \\ &= a_3x^3+(a_1+a_3)x^2+a_2x+(a_0+a_2)^2 \pmod{x^4+x+1} \\ &= (a_0b_0+a_3b_1+a_2b_2+a_1b_3) \\ &+ (a_1b_0+(a_0+a_3)b_1+(a_2+a_3)b_2+(a_2+a_1)b_3)x \\ &+ (a_2b_0+a_1b_1+(a_0+a_3)b_2+(a_2+a_3)b_3)x^2 \\ &+ (a_3b_0+a_2b_1+a_1b_2+(a_0+a_3)b_3)x^3 \pmod{x^4+x+1} \end{aligned} \tag{6.1}$$

Alternatively, this equation can be represented in matrix form as in Equation (6.2)

$$\begin{pmatrix} c_0 \\ c_1 \\ c_2 \\ c_3 \end{pmatrix} = \begin{pmatrix} a_0 & a_3 & a_2 & a_1 \\ a_1 & (a_0+a_3) & (a_2+a_3) & (a_2+a_1) \\ a_2 & a_1 & (a_0+a_3) & (a_2+a_3) \\ a_3 & a_2 & a_1 & (a_0+a_3) \end{pmatrix} \begin{pmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \end{pmatrix} \tag{6.2}$$

The circuit is implemented with 18 XOR and 16 AND gates. To scale this circuit up to $GF(2^{283})$ requires the calculation of a 283×283 matrix. This is easily calculated from the observation that the first column, $Z(0)$ is $A(x)$ and subsequent columns, $Z(i)$, are $x^iA(x) \pmod{x^4+x+1}$. From Table 6.3 it can be seen that although the parallel multiplier (mult_par) has an area of one magnitude larger than the serial approach (mult_lsb3), its latency and energy are considerable lower, leading to an area*energy*time figure that is two orders of magnitude less.

operation	area (gates)	area (mm^2)	time (μs)	time (cycles)	power (mW)	energy (nJ)	area*energy*time E-22mSJ
mult_lsb3	10217	0.025	1.776	222	2.33	4.14	1.8
mult_par	284322	0.674	0.008	1	34.62	0.28	0.01

Table 6.2: Comparison of bit-serial and bit-parallel $GF(2^{283})$ multiplier

Using the architecture for the Tate pairing accelerator outline in section 6.2, it is clear

that this would be a large device due to the presence of nine $GF(2^{283})$ multipliers, which are themselves larger than the overall Tate pairing architecture. In order to overcome this problem the tate_alu block has to be redesigned so that it only uses one $GF(2^{283})$ multiplier.

6.3.1 Multiplication in $GF(2^{283\times4})$ using mult_par

The technique outlined in Section 5.5.3 can be applied again to a multiplier in $GF(2^{283\times4})$ that uses the parallel multiplier in $GF(2^{283})$. Instead of nine parallel $GF(2^{283})$ multipliers, one multiplier is used in sequence to generate the intermediate values from equation (5.37) which are then stored in the registers such as rega3b3 (see Figure 6.5.

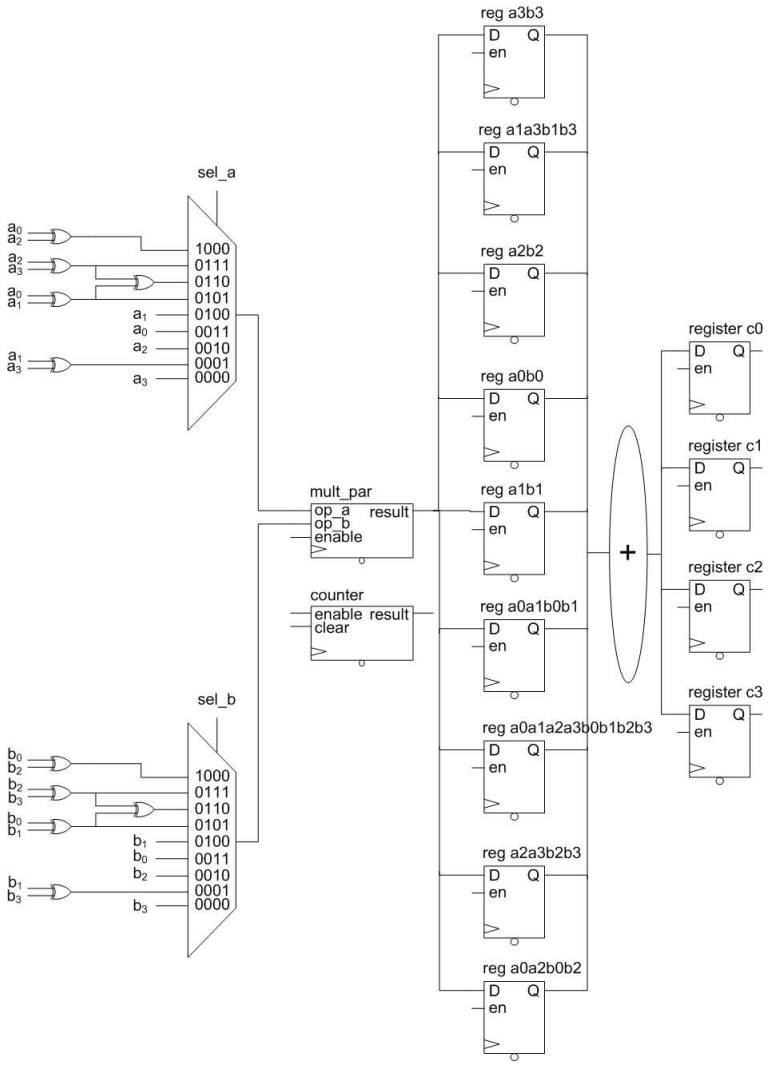

Figure 6.5: Datapath circuit for the parallel multiplier in $GF(2^{283\times4})$

The ASM charts for the multiplier and the control logic are shown in Figure A.15 in Appendix A.

From Table 6.3.1 it can be seen that although the new multiplier (mult_koa_par) has an area of one magnitude larger than the original approach (mult_koa), its latency and energy are considerable lower, leading to an area*energy*time figure that is two orders of magnitude less.

operation	area (gates)	area (mm^2)	time (μs)	time (cycles)	power (mW)	energy (nJ)	area*energy*time E-22mSJ
mult_koa	123059	0.295	1.926	241	22.55	43.4	247
mult_koa_par	393317	0.927	0.204	26	80.81	16.5	31.2

Table 6.3: Comparison of bit-serial and bit-parallel $GF(2^{283\times4})$ multiplier

6.3.2 Inversion in $GF(2^{283})$ using mult_par

An inverter can be designed using Fermat's little theorem and the parallel $GF(2^{283})$ multiplier. This necessitates a number of changes to the design as outlined in Section 5.5.7. In the datapath the output from the $GF(2^{283})$ multiplier feeds directly to the squaring circuit, whereas before it was registered (see Figure 6.6).

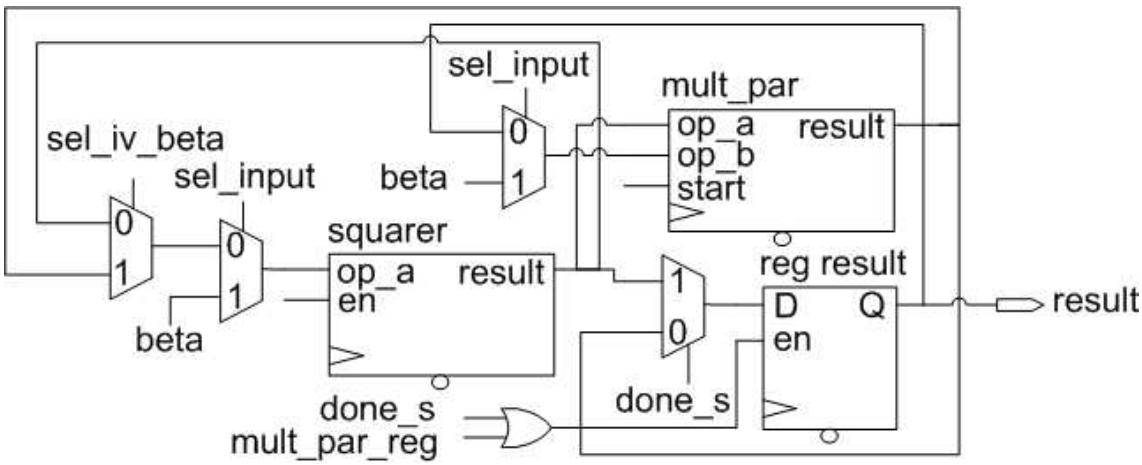

Figure 6.6: Datapath circuit for the inverter in $GF(2^{283})$using Fermat's little theorem and based on mult_par

In the ASM charts the decision box for mult_done has been removed as can be seen

in the ASM for the inverter and the control logic as shown in Figures A.17 and A.18 in Appendix A.

A comparison of the two inverters given in Table 6.3.2, leads to the conclusion that the original architecture for the inverter is the best choice.

operation	area (gates)	area (mm^2)	time (μs)	time (cycles)	power (mW)	energy (nJ)	area*energy*time E-22mSJ
inv_ferm	18412	0.044	22.65	2831	3.01	68.1	683
inv_ferm_par	314334	0.755	2.39	299	113.2	270	4870

Table 6.4: Comparison of bit-serial and bit-parallel $GF(2^{283})$ inverter

6.3.3 Inversion in $GF(2^{283\times4})$ using mult_par

Using the technique outlined in Section 5.5.7 and the previous designs outlined in this section the inverter in $GF(2^{283\times4})$ can be redesigned. The ASM charts are the same as in Section 5.5.7 are shown in Figures A.13 and A.14 in Appendix A.

The block diagram is modified to contain a sub-block called mkifm_alu, which preforms the operations of multiplications in both fields and inversion in $GF(2^{283})$. It inputs are muxed to the top-level so that the block shown in Figure 6.7 acts as the tate_alu in Figure 6.3.

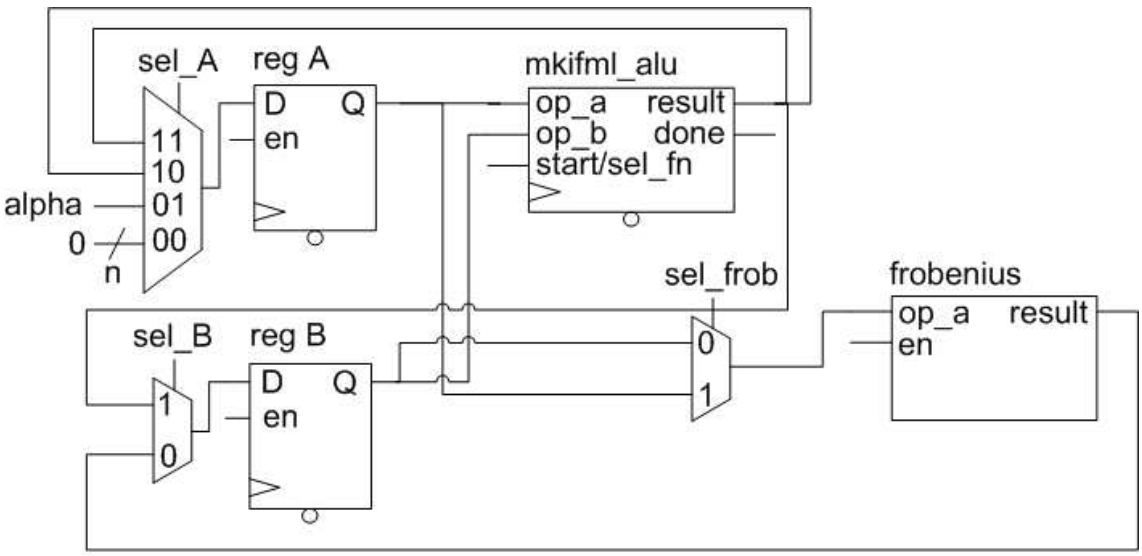

Figure 6.7: Datapath circuit for the inverter in $GF(2^{283\times4})$ based on mult_par

From Table 6.5 it can be seen that although the new inverter (inv_gf2m4_par) has an area of one magnitude larger than the original approach (inv_gf2m4), its latency and energy are considerable lower, leading to an area*energy*time figure that is 75% less.

Table 6.5: Comparison of bit-serial and bit-parallel $GF(2^{283\times4})$ inverter

operation	area (gates)	area (mm^2)	time (μs)	time (cycles)	power (mW)	energy (nJ)	area*energy*time E-22mSJ
inv_gf2m4	174048	0.417	29.863	3733	18.12	541	67500
inv_gf2m4_par	459522	1.085	3.98	497	97.87	389	16800

Table 6.6: Comparison of bit-serial and bit-parallel $GF(2^{283\times4})$ inverter

6.3.4 Experimental Results

The datapath and control logic for the block implementing the Tate pairing is the same as before (see Figures 6.4 and 6.3). The fastest that functional clock, tate_clk, can run at in this design is 125MHz. Results for the Tate pairing accelerator based on mult_par (tate_par) is shown in Table 6.3.4. From this table it can be seen that it has a latency of 0.08ms, an area of 1.139mm^2 and consumes 7660nJ of energy. Even though the area of this version of the pairing accelerator is 198% larger than the original, it still performs better when analysed with the area*energy*time metric. This is because the latency and energy figures are lower.

operation	area (gates)	area (mm^2)	time (μs)	time (cycles)	power (mW)	energy (nJ)	area*energy*time E-22mSJ
tate	250992	0.574	698.11	139622	42.4	29600	1.19E+08
tate_par	509267	1.139	79.77	9971	96.03	7660	6.96E+06

Table 6.7: Results for the Tate pairing accelerator

6.4 Evaluation

The hardware implementation compares very favourable with software. The running time in software is 177ms and it consumes 14.1mJ. In hardware these figures are 0.70ms and 0.0296mJ for the serial version, and a latency of 0.08ms and an energy consumption figure of 7660nJ for the version based on mult_par. Even though the area*energy*time metric is a useful tool for evaluating designs, there must be cut–off points for its various components. In area it has been decided that 300K NAND2 gate equivalents represents the area cut–off point. With this figure in mind, even though tate_par outperforms the tate architecture when evaluated against the area*energy*time metric, it will not be considered further due to its large area.

In terms of the overall system the inclusion of a hardware accelerator for the Tate pairing will lead to a 79% reduction in latency and energy when compared with running the protocol in software. Even with this improvement in performance, it is evident, with the protocol still taking 91.7ms and consuming 7.3mJ, that these percentage drops are not enough to justify the deployment of IBC in a WSN. Further significant improvements can be attained if the power operation (g^r) and the mult operation (rP) were also accelerated. Assuming that the same reductions in latency and energy consumption can be attained as for the Tate pairing calculation, then the figures are as in Table 6.4. As the time for

	no acceleration	Tate acceleration		Tate/$U = rP/g^r$ acceleration	
	value	value	% decrease	value	% decrease
energy	35.4mJ	7.26mJ	79.5%	0.08mJ	99.8%
time	444.5ms	91.7ms	79.4%	1.75ms	99.6%

Table 6.8: Effect of accelerating on IBC scheme

running the protocol is now 1.75ms and it consumes 0.08mJ of energy, then it is probable that the latency and energy figures for the protocol would be in appropriate range for a WSN. During data extraction from the network the device generating the symmetric key only receives a signature from the KGC. Therefore the only energy cost of the radio subsystem is the start-up energy cost of 440nJ (see Section 2.5.2).This means that running

the protocol requires $0.52mJ$. If there is a energy budget of $1000J$ and $1J$ is available for IBC, then the protocol can be run 1923 times.

At present there is not a large body of work with which to compare this contribution. There have been several examples of the Tate pairing implemented using Field Programmable Gate Array (FPGA) technology, but to the best of the author's knowledge this is the first attempt at an ASIC implementation, and it is also the first one where energy is considered as a key design goal.

As there are no ASIC implementations with which to compare these designs the smaller version of the design (tate) was ported to a Xilinx FPGA XC2VP100 device. The only change that was made to the original VHDL code was that the main bus was implemented using tri-states rather than multiplexers, in order to improve the performance of the circuit. The various contributions are detailed in Table 6.4. When compared with this work, account must be taken of the fact that this is a design targeting an ASIC that has been synthesised for a FPGA with minor modifications.

Shu et al [93] implement the Tate pairing using the η and η_T approach. They have also identified that the square root operation in $GF(2^{283})$ is very expensive and have redesigned their algorithm to not compute it. Also, they use digit serial multipliers in the field $GF(2^{283})$. Six of these are required for their design. For inversion in the field $GF(2^{283})$ they use the Itoh and Tsujii approach. Of the two approaches, η_T is faster than η when implemented in hardware, but at a considerable cost in area.

Ronan et al [55] have implemented a dedicated processor for the Tate pairing using the η_T approach. The square roots required in the algorithm are pre-computed and stored in a RAM. Pipelining is employed to improve the latency of the design. For multiplication in $GF(2^{283})$ a digit serial approach is employed and the Karatsuba algorithm is used for multiplication in $GF(2^{283\times4})$. Inversion in $GF(2^{283})$ is achieved using the modified EEA [15] and in $GF(2^{283\times4})$ the technique outlined in [60] is employed. Due to the complexity of the final exponentiation, it is implemented in a dedicated block. This architecture is larger and slower than Shu et al's version.

Keller et al [53] take the approach of designing an accelerator that can perform the Tate

pairing calculation and also elliptic point scalar multiplication. They use the BKLS/GHS algorithm, and note that it can be altered to perform point multiplication by removing the steps in $GF(2^{mk})$. It has dedicated blocks for calculating the line function and point addition and doubling on the elliptic curve. In $GF(2^{283})$ multiplication is performed using a digit serial approach and the divisions are achieved using the techniques detailed by Shantz [91]. The Karatsuba algorithm is used for multiplication in $GF(2^{283\times4})$. They achieve a latency of 2810 μs, whilst maintaining a relatively small area.

Keller et al [52] develop their work further by the inclusion of a dedicated $GF(2^{283\times4})$ inverter block. This inverter block is based on the algorithm presented in [60], and it requires thirty-six $GF(2^{283})$ multiplications and one inversion in $GF(2^{283})$. By including this block they find that they improve the latency to 1670 μs but increase the area by 34%.

Ronan et al [81] use a genus two hyper-elliptic curve defined over $GF(2^{106})$ to implement the Tate pairing. As this curve has a security multiplier of twelve, its level of security is comparable with the elliptic case defined over $GF(2^{283})$. The architecture is divided into four blocks. A per-computation block calculates and stores squares of the in-out points. An ALU performs all the operations required in $GF(2^{106})$ and $GF(2^{106\times12})$. There is also a controller block and a memory block. Multiplication in the smaller field is performed using a digit serial approach, and in the large field using Karatsuba algorithm. Inversion in $GF(2^{106})$ and $GF(2^{106\times12})$ is achieved using the modified EEA of [15] and [60] respectively. This architecture has an area of 43986 slices and 749 μs

There are a couple of implementations in characteristic three. Kerins et al [54] use the Duursma–Lee to calculate the Tate pairing on $E(3^{97})$. Their architecture has an area of 50286 slices and a latency of 5920 μs. Grabher and Page [35] also implement the Tate pairing in characteristic three. Instead of having a hard-wired solution they use a general purpose processor to control the an ALU co-processor. This enables the calculation of the pairing and also that of the elliptic curve point multiplication required by IBC. It is unclear how the stated timings in the paper are arrived at. Given that the ALU has an area of 1700 slices and the multiplier in $GF(3^{97})$ is 946 slices, it is clear that this device can only contain one multiplier, which requires 28 clock cycles to perform the multiplication

and two cycles to fetch and store the result. If the top frequency of 150MHz is considered, then one multiplication takes 210ns. And as there are, at least, 20 multiplications in a loop and 97 loops, then, discounting other operations, the Tate pairing will take 407μs. This figure is larger than 399μs quoted in their paper.

From an analysis of the above designs, it is apparent that a digit serial approach to multiplication in $GF(2^{283})$ should be investigated. If a digit serial approach was employed it would offer no benefit in terms of performance if the square root operation's latency was not reduced. Therefore the square root could be precomputed and stored in RAM as in [55]. It is estimated that if this technique was used in tate_par that the latency would decrease to 75μs. Some of the implementations use the technique outlined in [60] for inversion in $GF(2^{283\times4})$. This requires the same number of multiplications in $GF(2^{283})$ as the approach used in this thesis, and so it offers no advantage.

As can be seen from Table 6.4, this implementation is quite slow when compared with the work of Shu et al [93]. It is thirty six times slower but only 20% smaller. It is believed that the slice count could be greatly reduced by the use of block RAM in lieu of the registers in this design. Also, tri–states could be more widely used to increase the speed of the circuit.

design	curve	algorithm	Xilinx device	CLB Slices	frequency (MHz)	Time (μs)
tate	$E(2^{283})$	η	XC2VP100	29929	59	2360
[93]	$E(2^{283})$	η	XC2VP100	22726	84	87
[93]	$E(2^{283})$	η_T	XC2VP100	33252	56	78
[55]	$E(2^{313})$	η_T	XC2VP100	41078	50	124
[53]	$E(2^{283})$	BKLS/GHS	XC2V6000	24655	47	2810
[52]	$E(2^{283})$	BKLS/GHS	XC2V6000	33047	33	1670
[81]	$E(2^{103})$	η	XC2VP125	43986	32	749
[54]	$E(3^{97})$	Duursma–Lee	XC2VP125	50286	19	5920
[35]	$E(3^{97})$	Duursma–Lee	XC2VP4	4481	150	399

Table 6.9: Comparison with prior art

6.5 Summary of Contributions

In this chapter a low cost Tate pairing accelerator is designed and implemented in a TSMC 65nm low power process. Using the η algorithm, an architecture for implementing the Tate pairing calculation is arrived at. An analysis of the algorithm led to the decision that two ALUs should be incorporated in the datapath. There are two variations of this architecture designed; one is based on a bit serial multiplier and the other is based on a parallel multiplier. For reasons of area, it is considered that the serial version is the most appropriate choice for inclusion in a wireless sensor node.

The Tate pairing in hardware takes $0.7ms$ to complete and consumes $29.6\mu J$ of energy, as compared to $177ms$ and 1.41E-2J when run on an ARM machine. This leads to a significant reduction in the cost of running IBC on a constrained device. It has been discussed how with the acceleration of the other computationally demanding components of this algorithm that it would be possible to use IBC in the context of a WSN.

Due to a lack of ASIC designs with which to compare this design, it was ported to a Xilinx FPGA XC2VP100 device to enable comparisons with existing solutions. Unfortunately, the prior art outperforms this design in terms of area and latency, but this is a rather unfair comparison, as account must be taken of the fact that this implementation of the η pairing targets an ASIC.

CHAPTER 7

Conclusions & Future Work

7.1 Conclusions

7.1.1 Motivation for Proposed Research – A Summary

Developments in CMOS radio, electronics and MEMS technologies are enabling the convergence of three key technologies onto the one platform. These are sensing, computation and RF communication. The emergence of these devices will enable sensing technologies to become embedded in our environment, to interact with us in new and meaningful ways. Some of areas in which they devices could be deployed are smart roads, healthcare and environmental monitoring.

Different application areas will give rise to different challenges, but there is a common set of design constraints that a WSN device must address. One of the main drivers for adoption of this technology will be cost. At present, large sensors, such as the AreaRAE IAQ from RAE [78], are very expensive. If they could be replaced by cheap sensors, that even though they might not be as sensitive, are able to achieve the same level of result, then this would encourage deployment of WSNs. It will be required, again for reasons of cost, that these networks have a long lifetime on a very limited energy supply. A figure of the order of $1000J$ is the energy resource of these devices, and therefore it is imperative to reduce energy dissipation from the network level down to the algorithms that run on

the devices. There is no point having a low cost network if it is costly to deploy and maintain, so it must be self-configuring and robust to individual wireless sensor node failure. In most deployments of these devices, security of the data in the network will be a requirement.

A symmetric key cryptosystem is appropriate for communication between the nodes, though a pre-installed system wide symmetric key or pair-wise keys stored on the devices are not suitable for reasons of security and lack of memory, respectively. Therefore an asymmetric or public key system is required to establish the symmetric keys between individual nodes. This thesis is concerned with finding a solution to the key distribution problem in a WSN.

7.1.2 Summary of Thesis Contributions

The concept of a low cost wireless sensor network is discussed in chapter 1. Of the example application areas discussed in this chapter; environmental monitoring, in the context of a counter-terrorism role, is highlighted as the target application. This is because of the high level of security required.

Chapter 2 outlines the technical challenges facing the emerging technology of WSNs. Along with cost, energy dissipation and ease of use; security is highlighted as a key challenge to make the vision of ubiquitous sensing a reality.

Chapter 3 is an overview of the mathematics required for understanding of the research in the later chapters. The Tate pairing is introduced and various algorithms for calculating the pairing are discussed. It has been concluded that the the η algorithm [59, 6] would be the most amenable to a low energy hardware implementation. This is because it has the least number of multiplications in $GF(2^{283})$ and has a regular structure that maps well to hardware.

In chapter 4 the security requirements of a WSN are discussed. It is concluded that a symmetric scheme is the appropriate choice for securing data in the network, and that a system based on IBC should be implemented to distribute the symmetric keys. The BMLQ IBS and SOK identity-based NIKDS schemes are used to develop a protocol that

authenticates network access, distributes local symmetric keys and provides a mechanism for the WSN to communicate securely with a sink. It is evaluated against well known attacks on a WSN and found to perform well. The component parts of the protocol are profiled in terms of latency and energy consumption. It was decided that the most computationally intensive part of the scheme, which is the Tate pairing, would be implemented in hardware.

Chapters 5 and 6 describes the design of a low cost Tate pairing accelerator to be implemented in a TSMC 65nm low power process. The η algorithm is chosen to implement the Tate pairing. This algorithm is analysed for the various arithmetic operations in the fields $GF(2^{283})$ and $GF(2^{283\times4})$ that are required, and these operations are designed and then evaluated in terms of their area, energy and latency. A top level architecture is designed which incorporates these arithmetic units in two ALUs. Two variations of the Tate pairing accelerator are designed; one based on a bit-serial LSB $GF(2^{283})$ multiplier and another based on a parallel $GF(2^{283})$ multiplier.

Timing and energy figures for the accelerators are generated and compared with software. From the area of the two approaches it is clear that the accelerator based on the bit-serial multiplier is the best choice for a WSN. An estimation is made for the total energy and time required for implementing the protocol when the exponentiation and point multiplication functions are also accelerated on hardware. These figures prove that a system based on IBC could be considered for distributing symmetric keys in a WSN. The design is ported to an FPGA and compared against existing solutions.

7.1.3 Research Objectives Achieved

The objectives of this research objectives as set out in Section 1.4 have been achieved as follows.

1. Well known techniques for implementing security in a WSN were evaluated. The key considerations were; the security offered by the protocol, ability to scale and amounts of bits required to be transmitted by the radio, which affects the energy consumed in the network. The conclusion was reached that a symmetric scheme,

such as AES is warranted, but that a scheme based on IBC should be used to distribute the symmetric keys.

2. A protocol that uses BMLQ IBS and SOK identity-based NIKDS has been proposed to distribute symmetric keys in a WSN. It uses the fact that the wireless sensor nodes will know their relative position to provide protection against well known attacks. The protocol was profiled on an ARM platform [5] in order to measure the energy consumption of its component parts, timing figures were also obtained. The Tate pairing was identified as consuming the most energy.

3. The Tate pairing has been designed using a TSMC 65nm low power process. The Miracl library was used to verify the functionality of the Tate pairing accelerator.

4. The approach has been validated in that the hardware implementation greatly reduced the amount of energy and time required to run the Tate pairing when compared with software. There are no ASIC implementations with which to compare this work, but it has been compared with FPGA solutions.

7.2 Future Work

The next natural stage of this work would be the implementation of elliptic curve multiplication and exponentiation in the field $GF(2^{283\times4})$ in order to prove the figures given in Table 6.4. This could be achieved using the hardwired approach outlined in this thesis i.e. there could be a series of FSMs present for implementing the different algorithms. A better approach would be to make the device programmable. As well as enabling the two operations mentioned above to be performed, a programmable device would also allow the different algorithms that calculate the Tate pairing, such as the η_T method, to be run.

The block implementing the Tate pairing calculation could be altered so that instead of a FSM generating the control logic, there would be a FSM generating an address. This five bit address would be used to control a multiplexer which select the relevant control logic. The control logic required is 32 bits wide and is stored in a register bank. In total

there would be 32 of these 32 bit registers. They are written via the APB interface. The only operation that is required to be added to the datapath is the squaring operation in $GF(2^{283\times4})$.

Consideration should also be given to implementing the improvements that were discussed in Section 6.4, such as investigating the benefits of a digit serial approach to multiplication in $GF(2^{283})$ and precomputation of the required square roots.

Now that it has been determined that a IBC scheme could be implemented on a wireless sensor node, if it is accelerated in hardware, then the protocol as outlined in section 4.7 can also be used in WSN. This protocol now has to be validated, firstly in a software simulation environment, and then in a real depolyment. Some of the characteristics of a WSN that need to be modeled are; the energy dissipation of the devices by the radio and digital computation, the routing algorithm that are used to establish the network and the protocol given in this thesis. This model could be used to evaluate the protocol against the attacks listed in Section 4.8.

Appendices

APPENDIX A

Algorithmic State Machines for the designs

A.1 Introduction

In this appendix the ASM diagrams for the various circuits in Chapters 5 and 6 are presented.

A.2 Algorithmic State Machine Diagrams

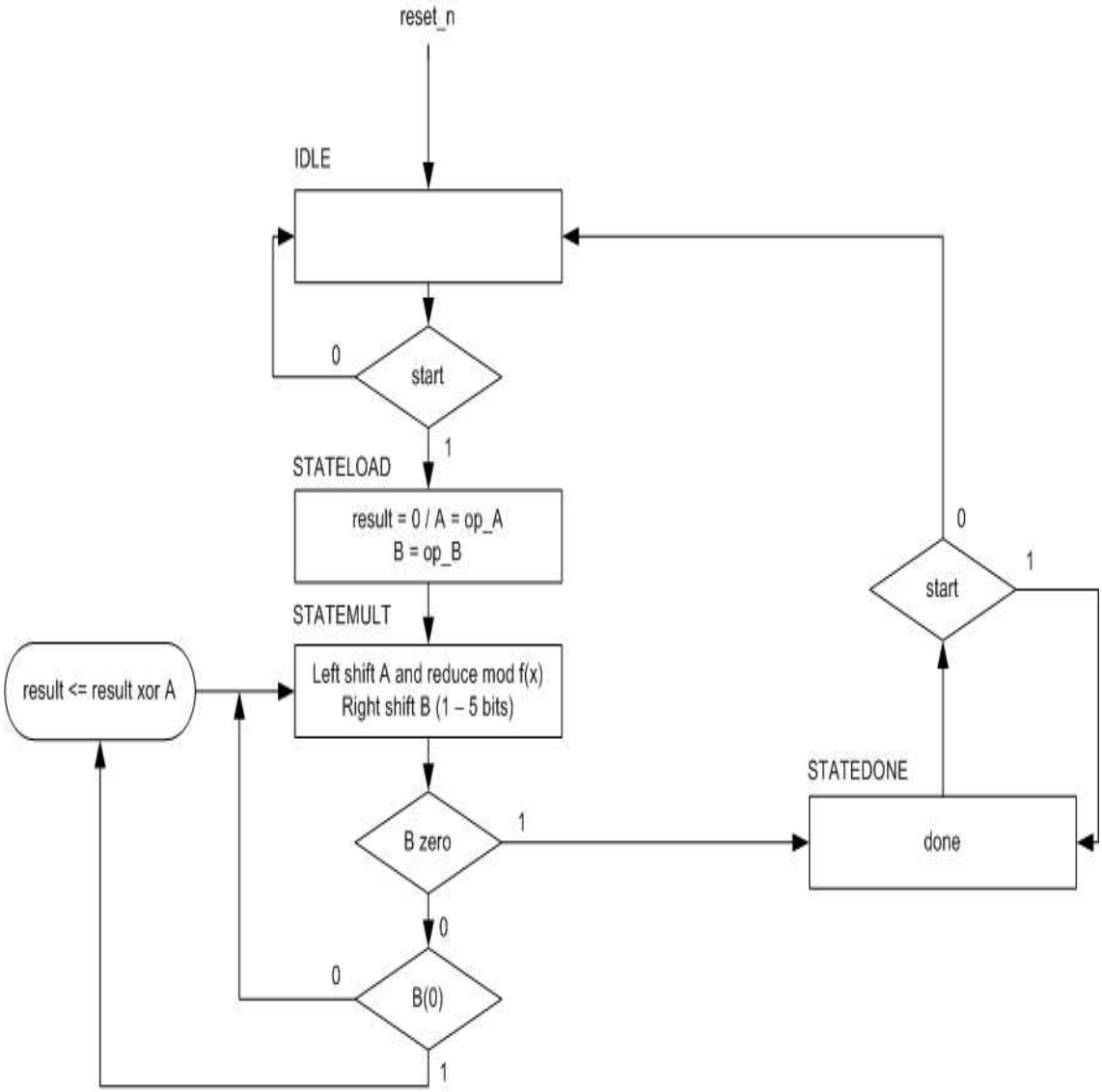

Figure A.1: ASM chart for the lsb multiplier in $GF(2^{283})$

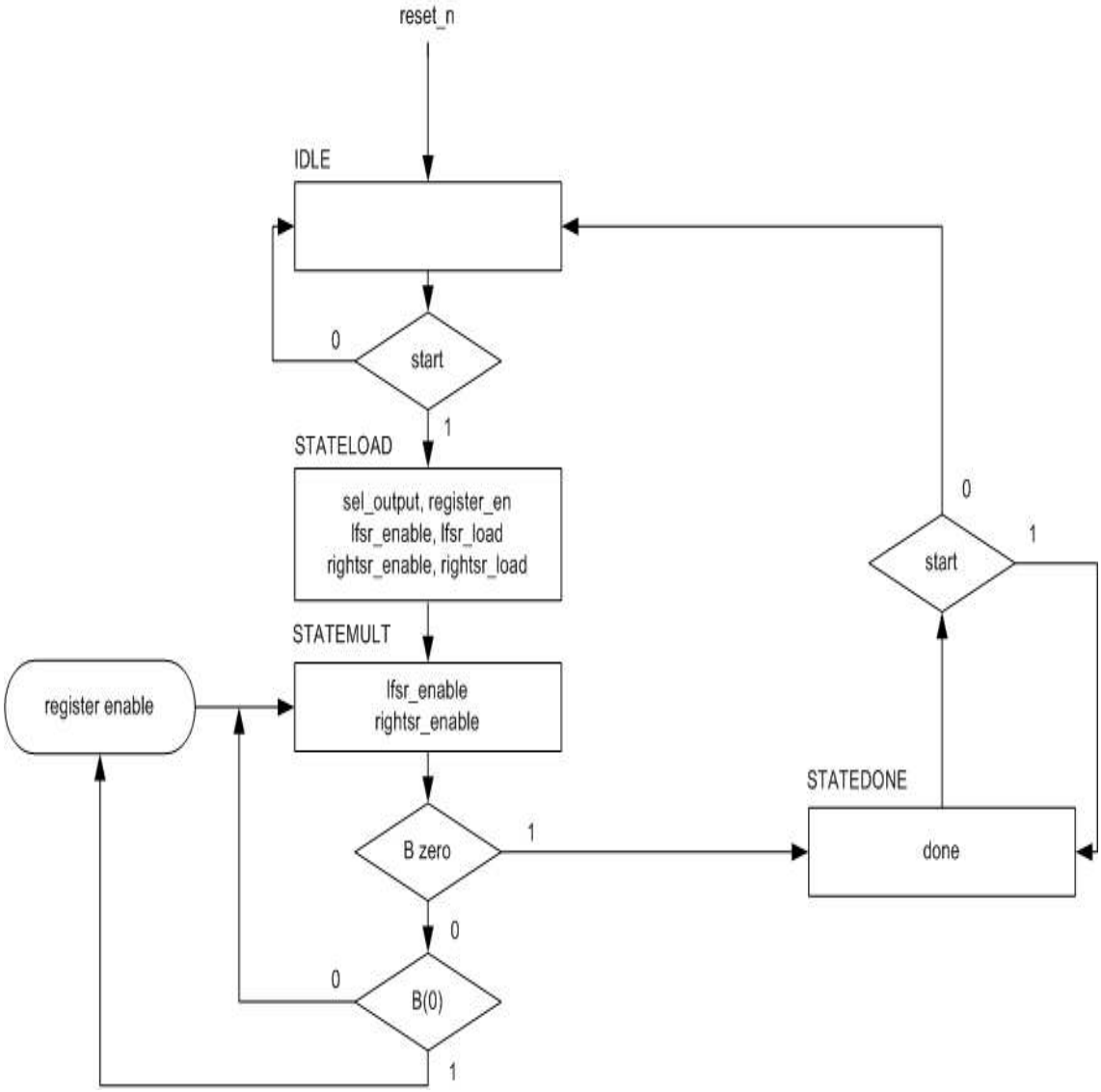

Figure A.2: ASM for the control circuit for the lsb multiplier in $GF(2^{283})$

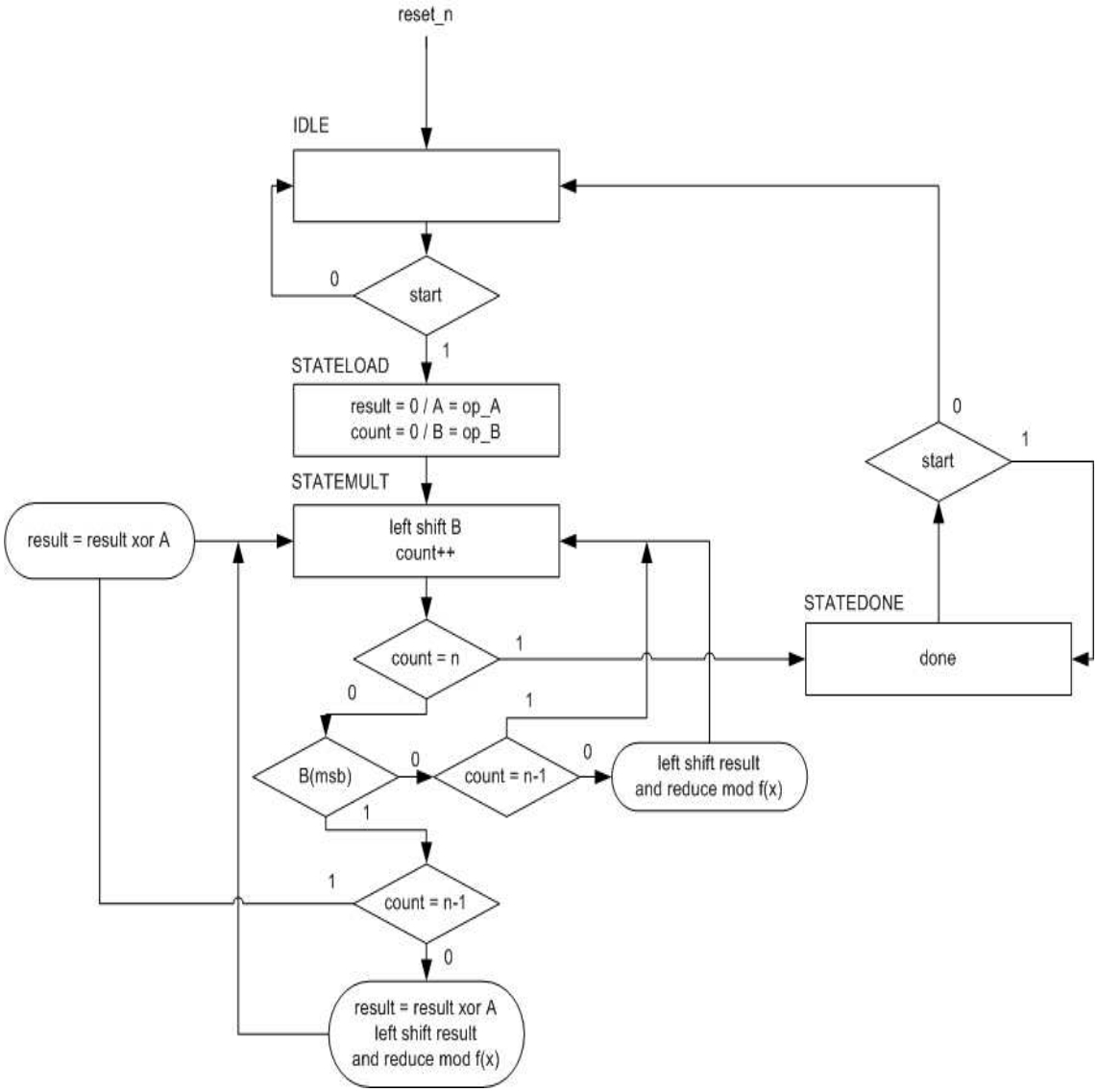

Figure A.3: ASM chart for the msb multiplier in $GF(2^{283})$

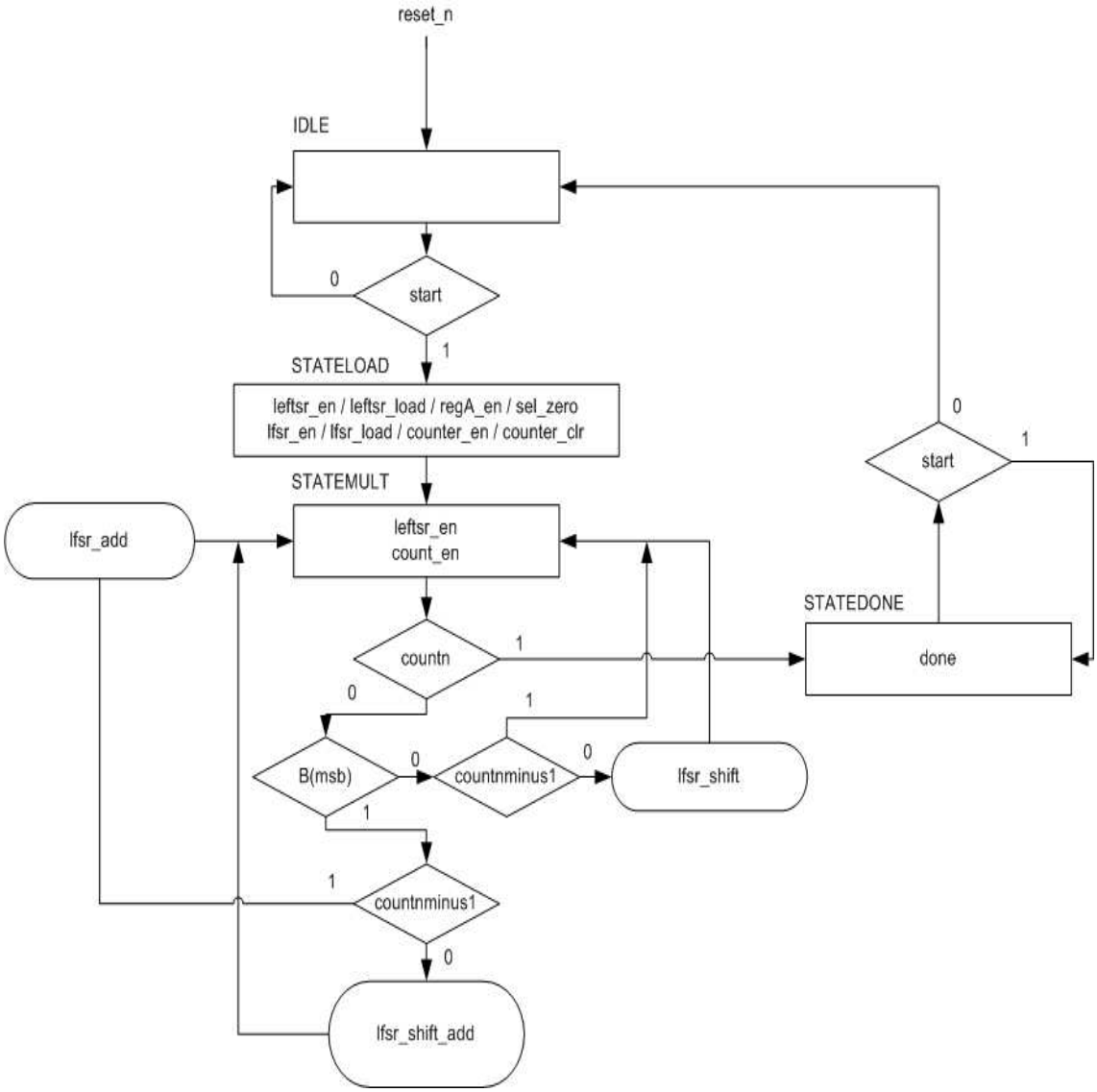

Figure A.4: ASM for the control circuit for the msb multiplier in $GF(2^{283})$

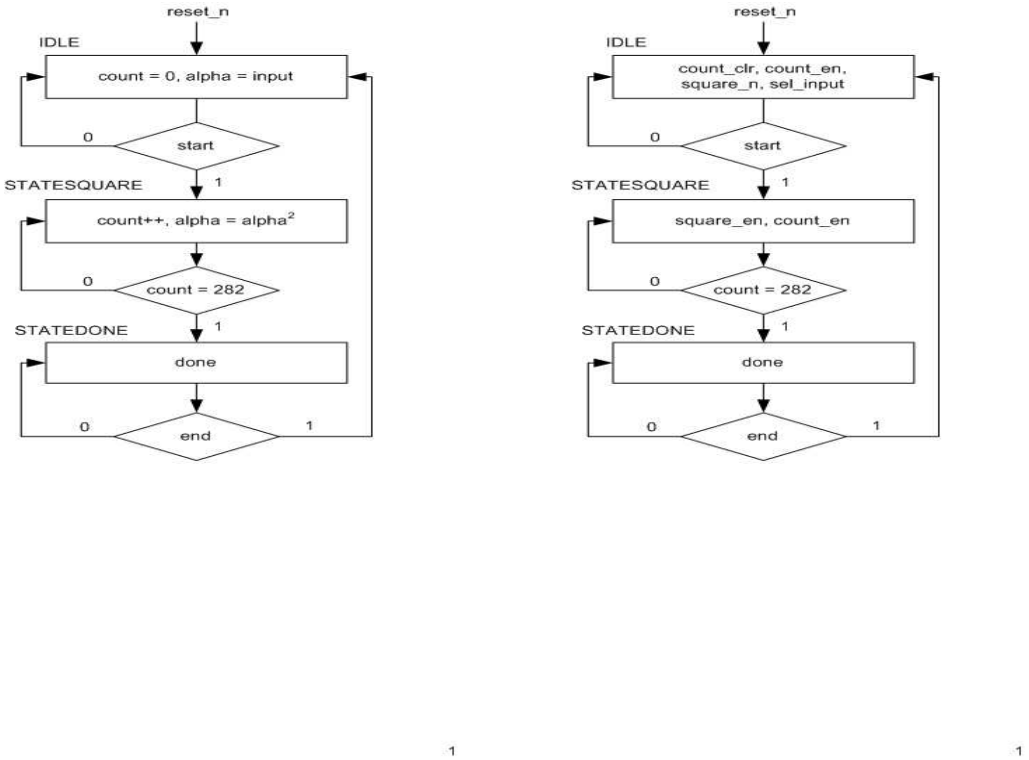

Figure A.5: ASM charts for the serial square root operation in $GF(2^{283})$

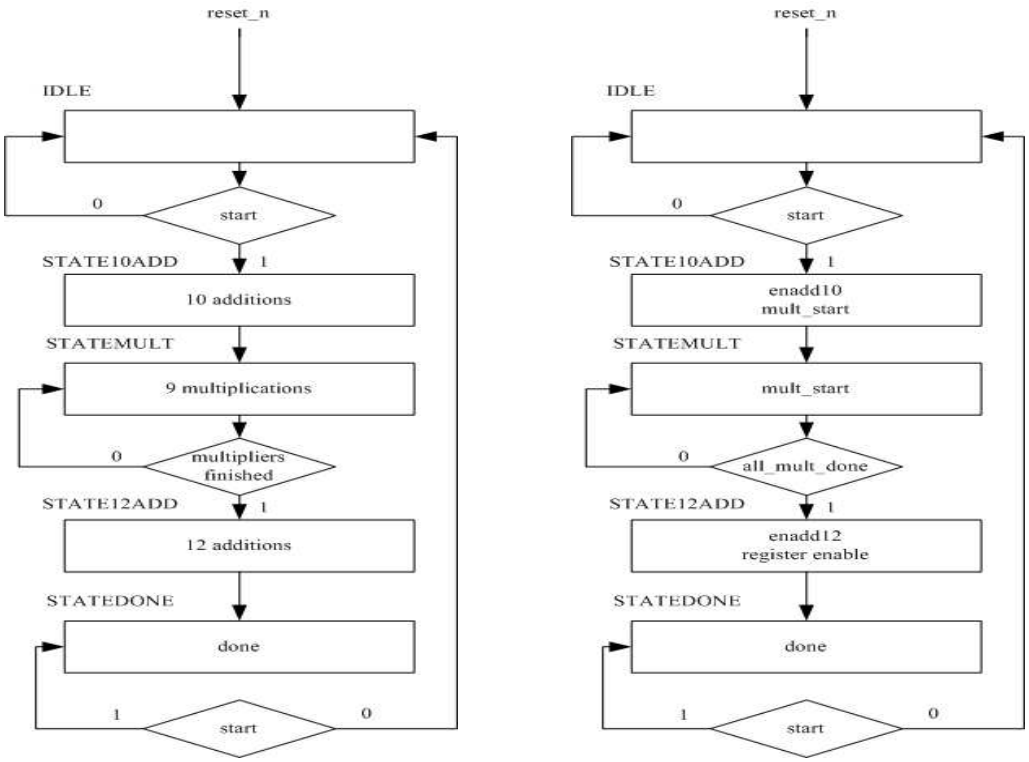

Figure A.6: ASM charts for the multiplier in $GF(2^{283\times4})$

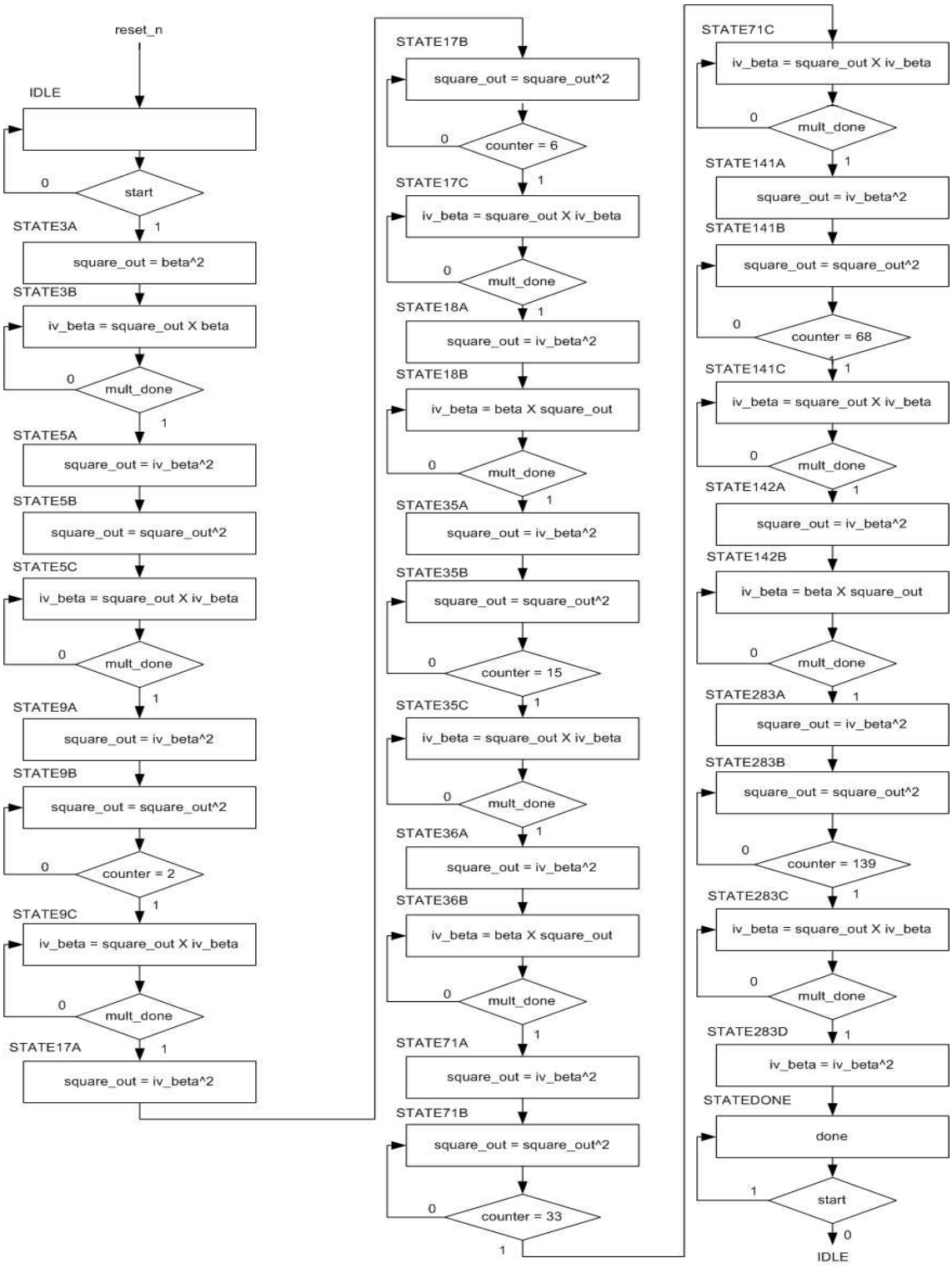

Figure A.7: ASM chart for the inverter in $GF(2^{283})$ using Fermat's little theorem

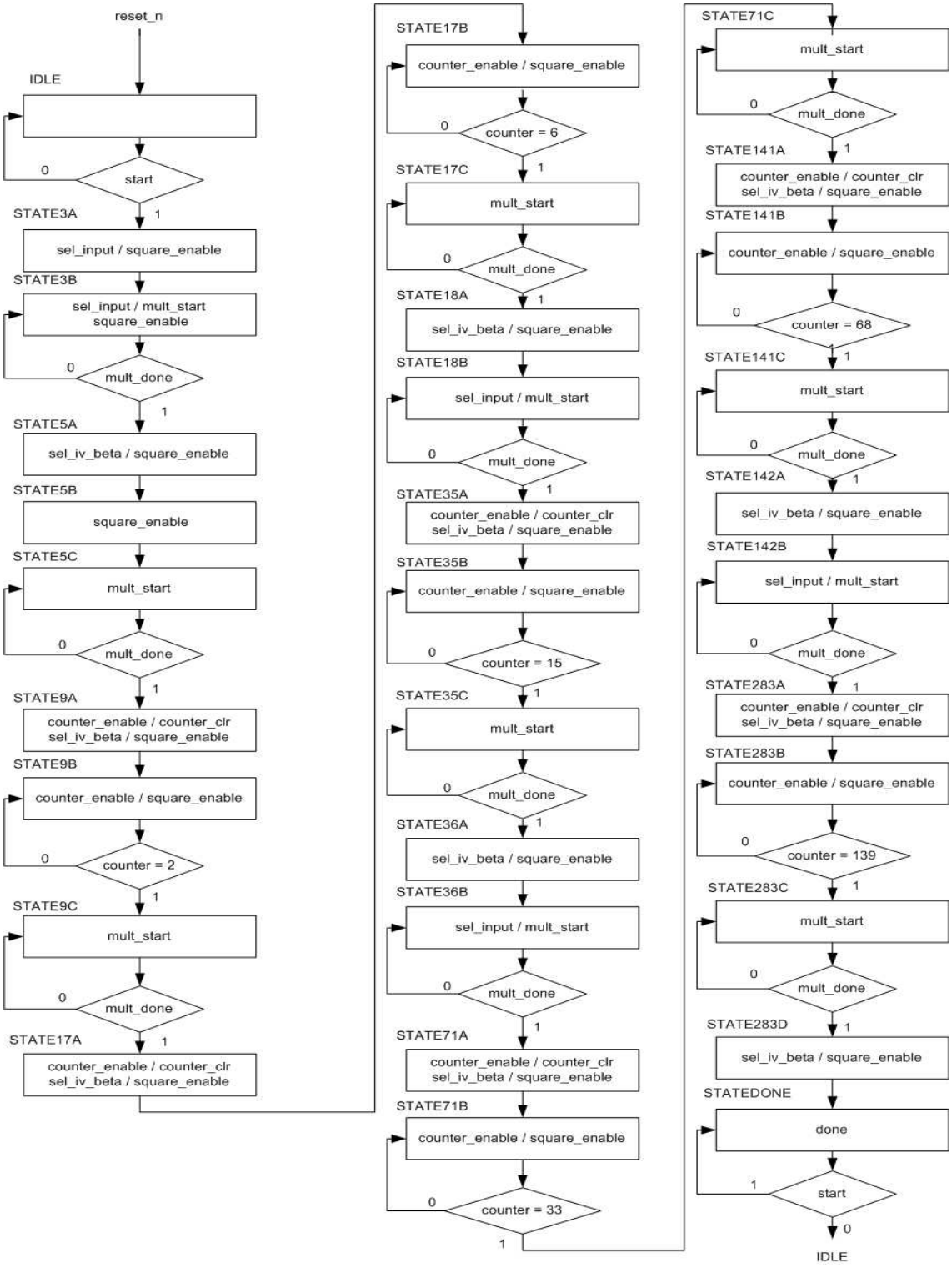

Figure A.8: ASM for the control circuit for the inverter in $GF(2^{283})$ using Fermat's little theorem

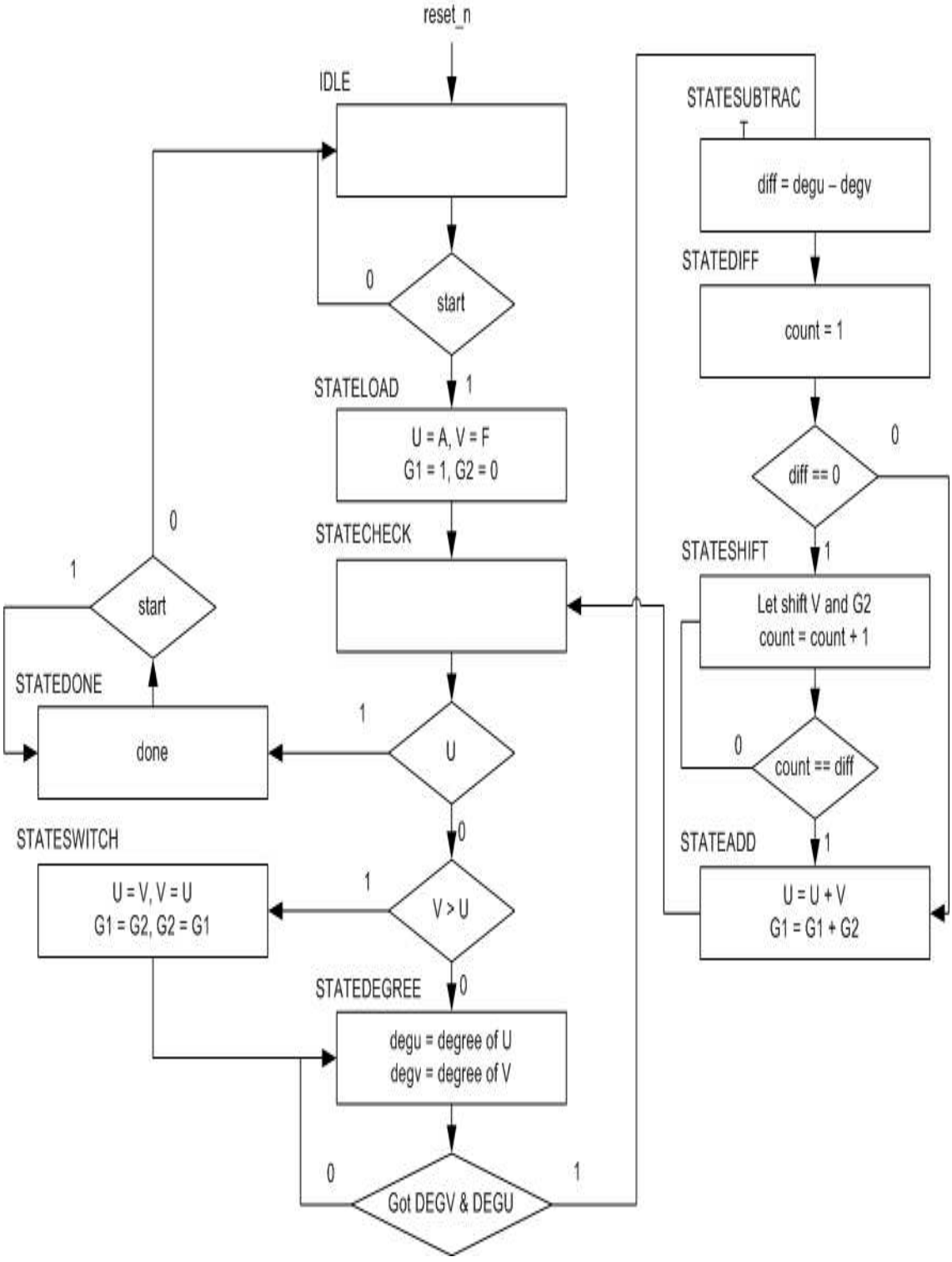

Figure A.9: ASM chart for the inverter in $GF(2^{283})$ using the Extended Euclidean Algorithm

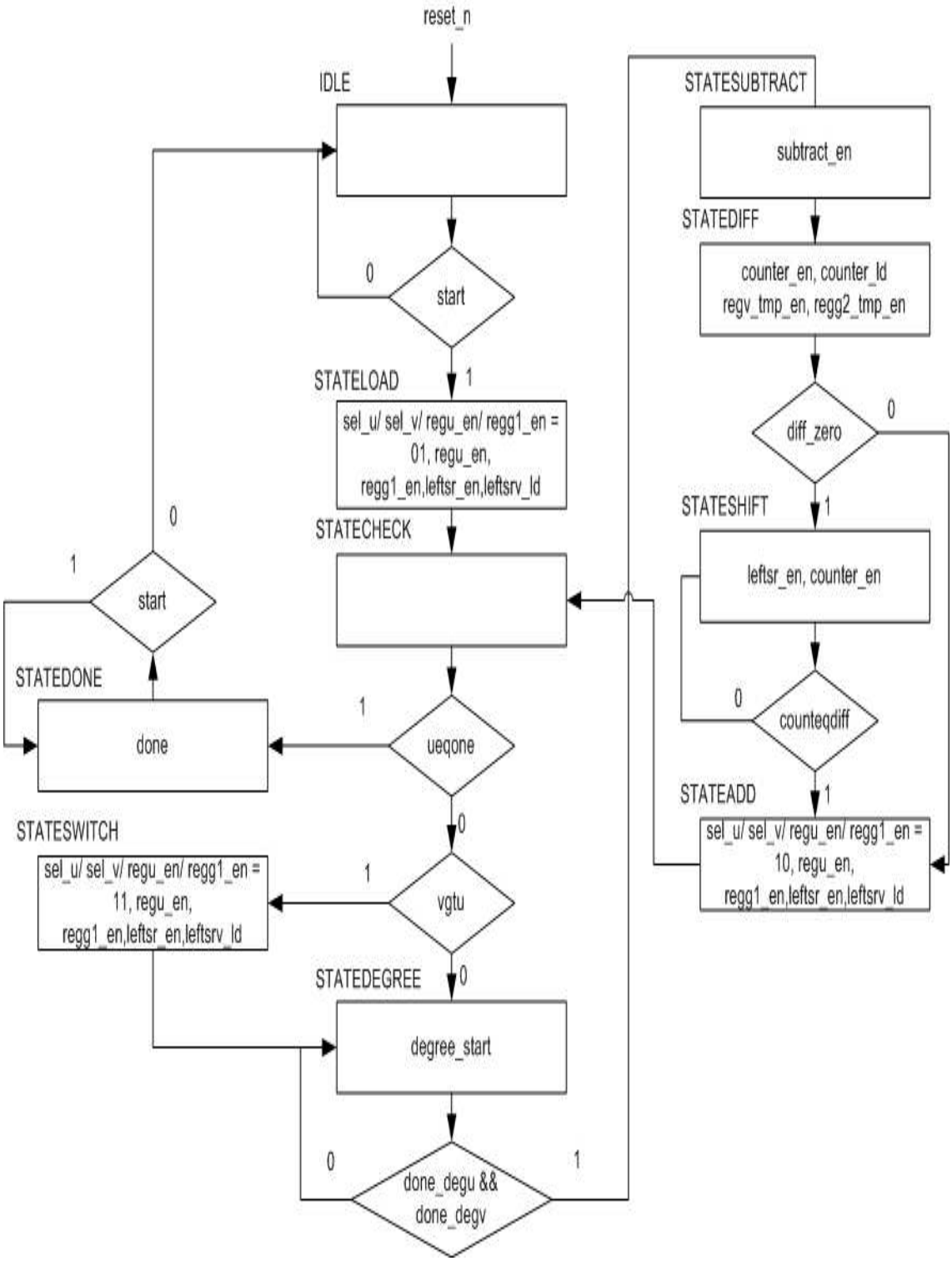

Figure A.10: ASM for the control circuit for the inverter in $GF(2^{283})$ using the Extended Euclidean Algorithm

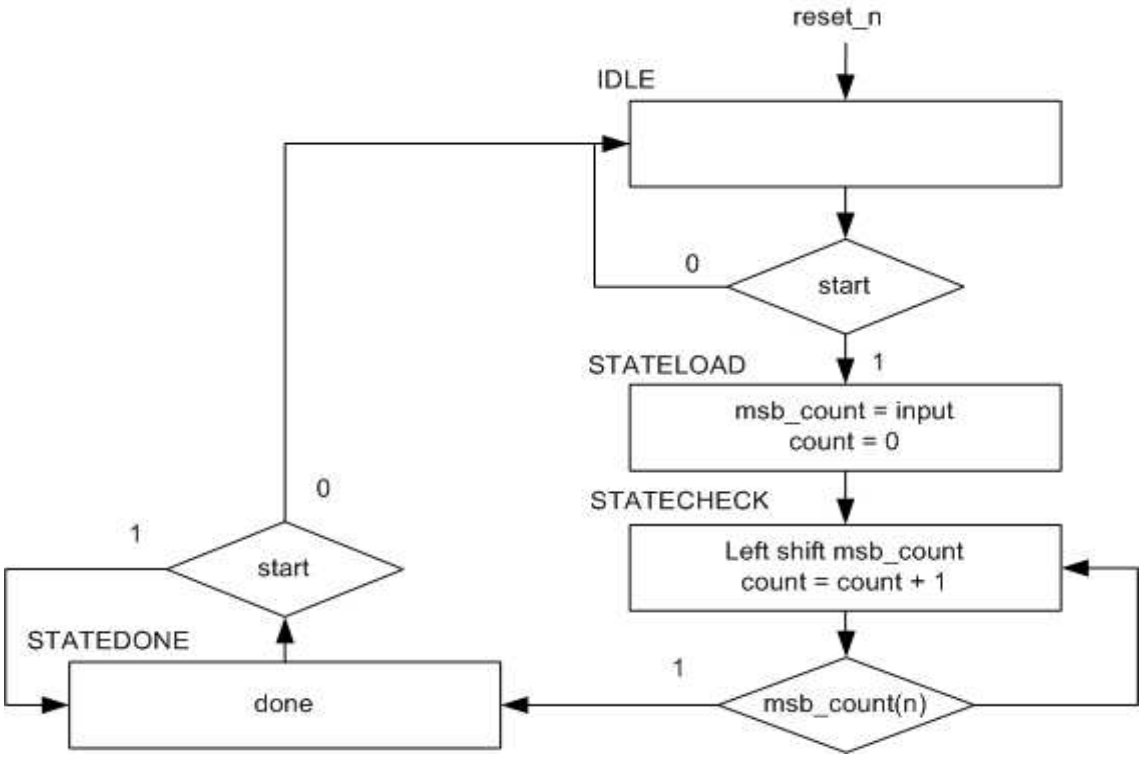

Figure A.11: ASM for getting the degree of a polynomial

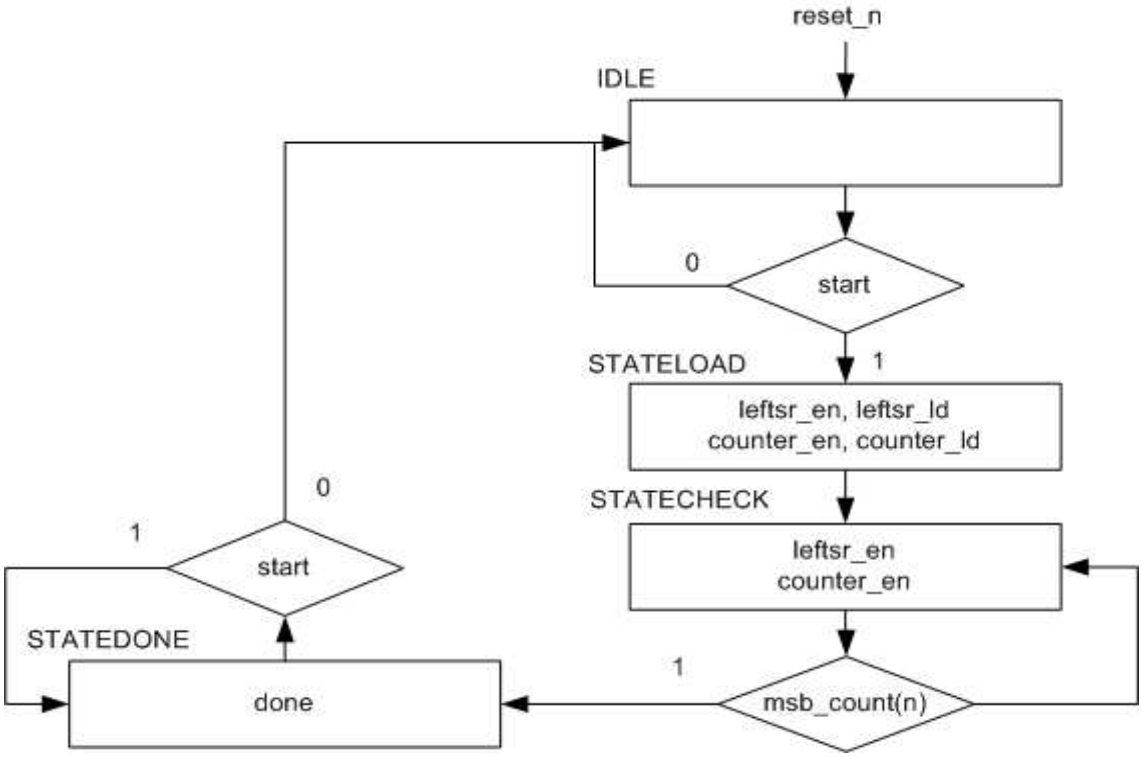

Figure A.12: ASM for the control circuit for getting the degree of a polynomial

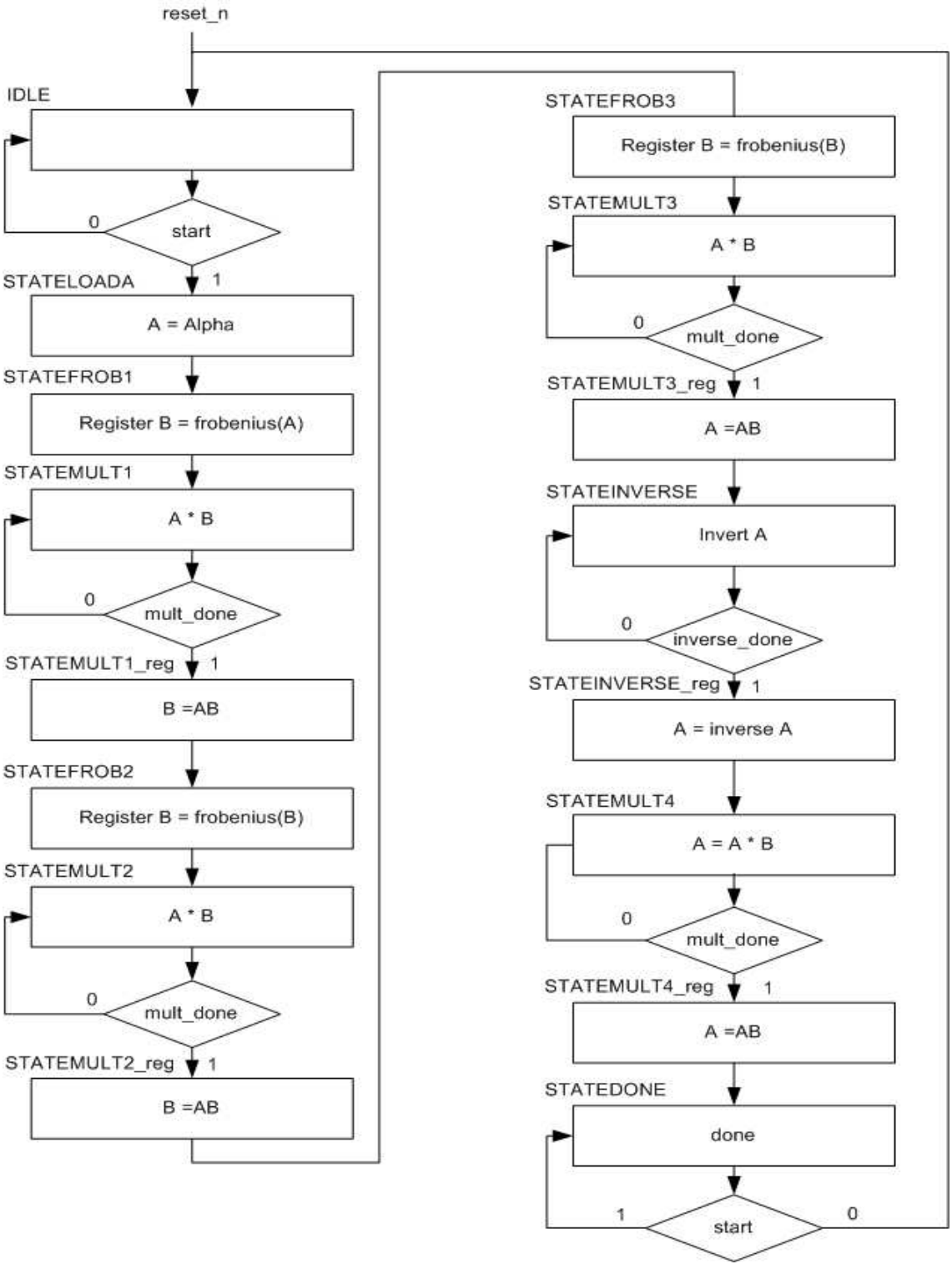

Figure A.13: ASM chart for the inverter in $GF(2^{283\times4})$

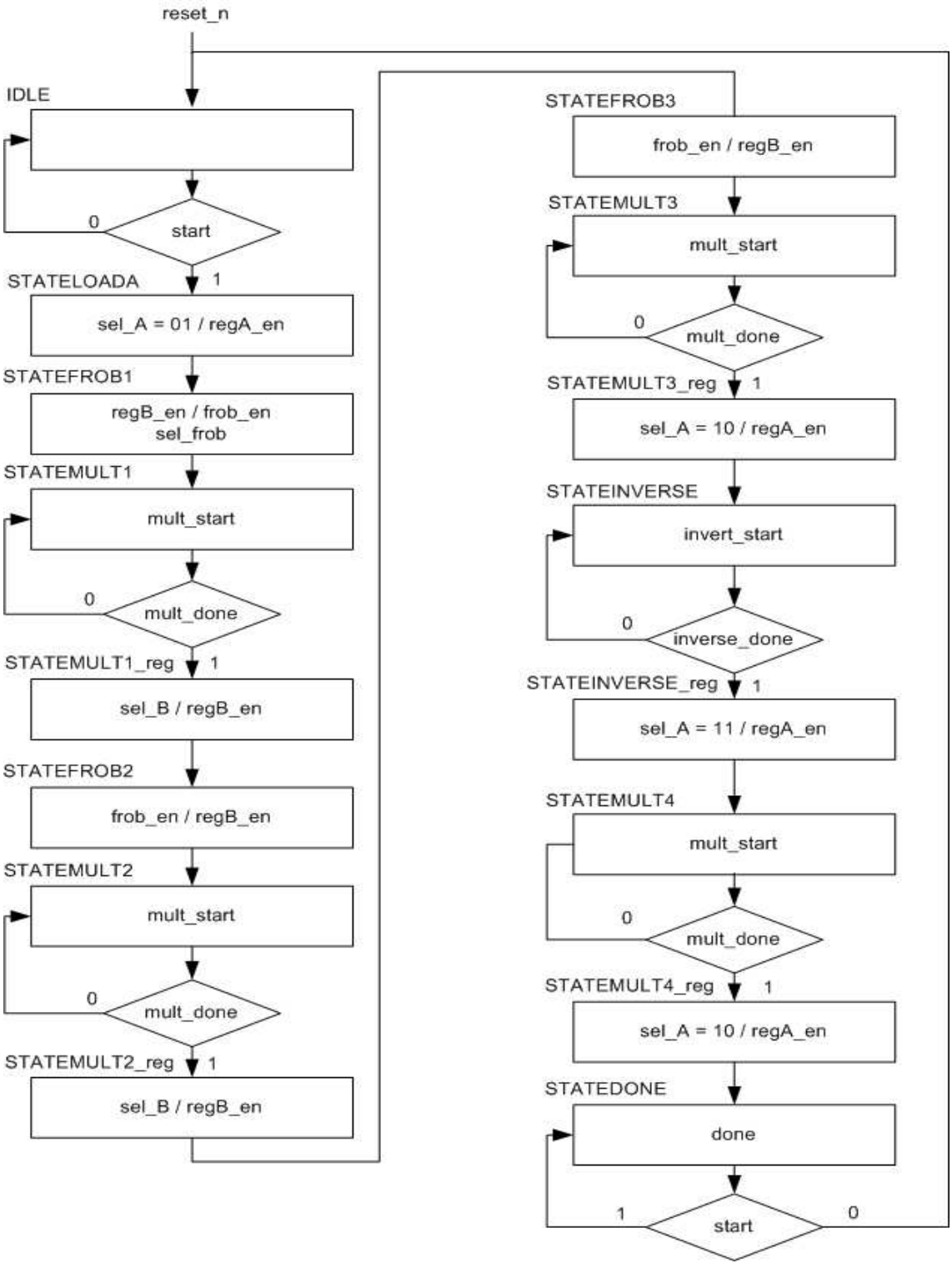

Figure A.14: ASM for the control circuit for the inverter in $GF(2^{283\times4})$

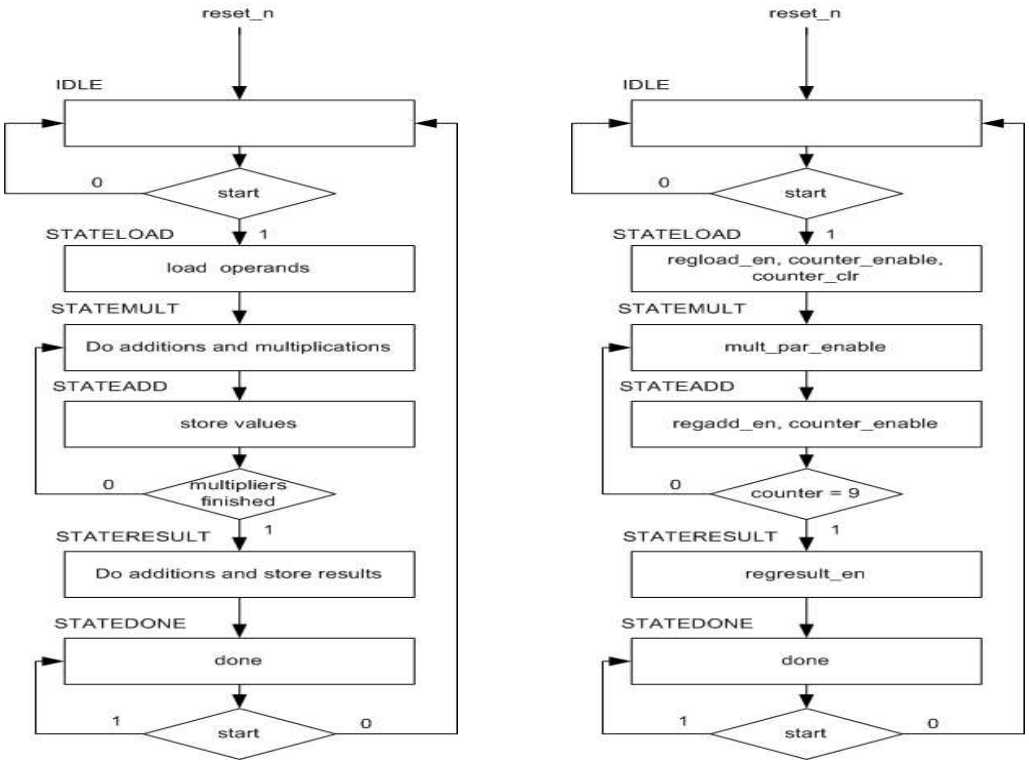

Figure A.15: ASM charts for the parallel multiplier in $GF(2^{283\times4})$

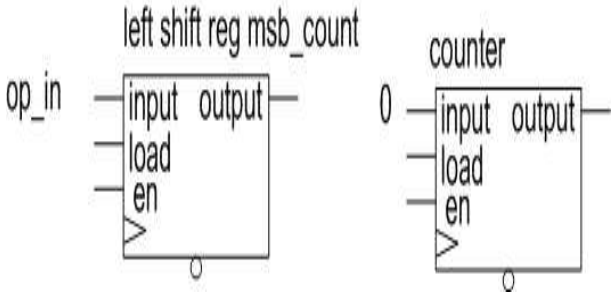

Figure A.16: Datapath circuit for getting the degree of a polynomial

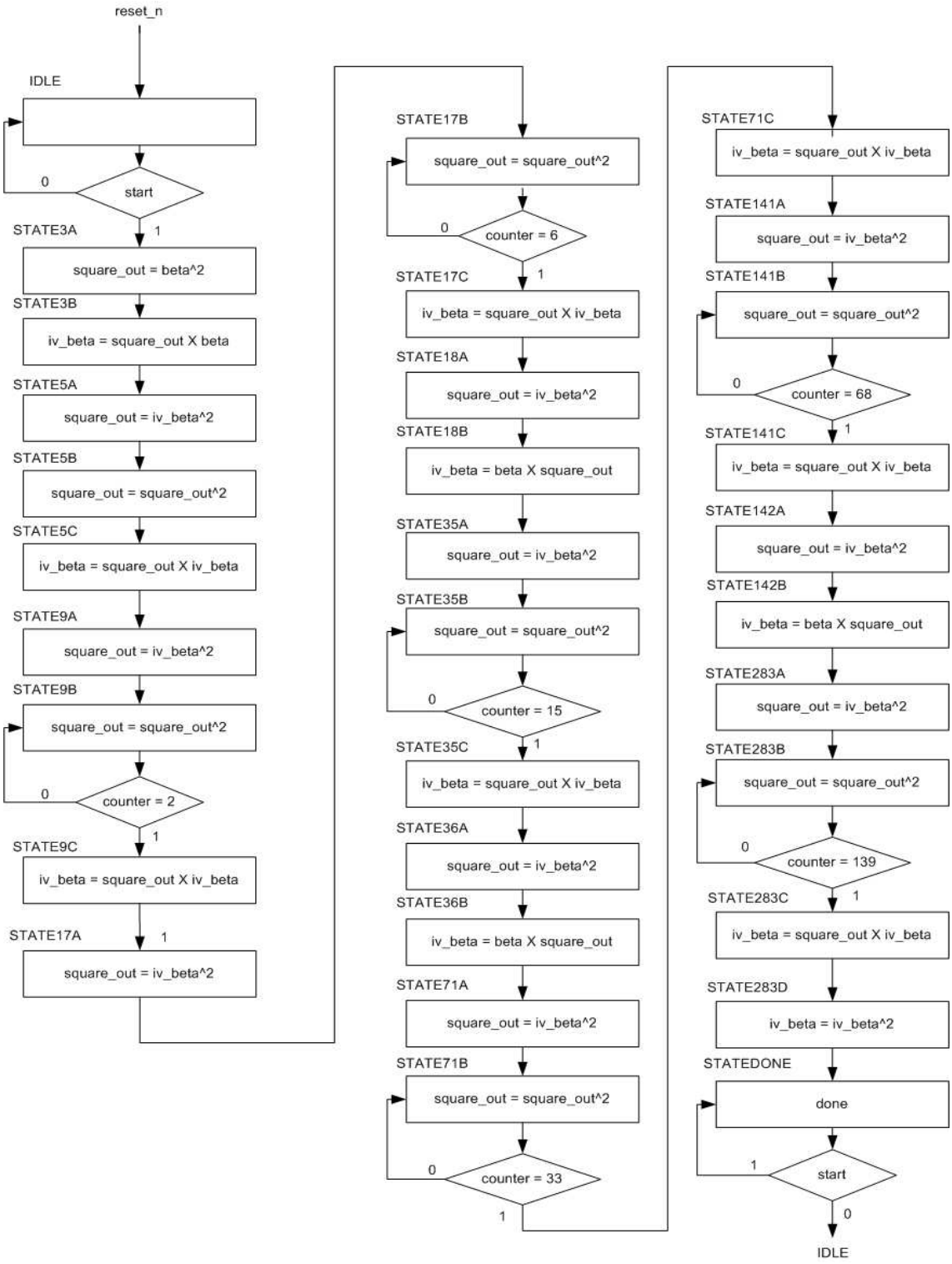

Figure A.17: ASM chart for the inverter in $GF(2^{283})$ using Fermat's little theorem and based on mult_par

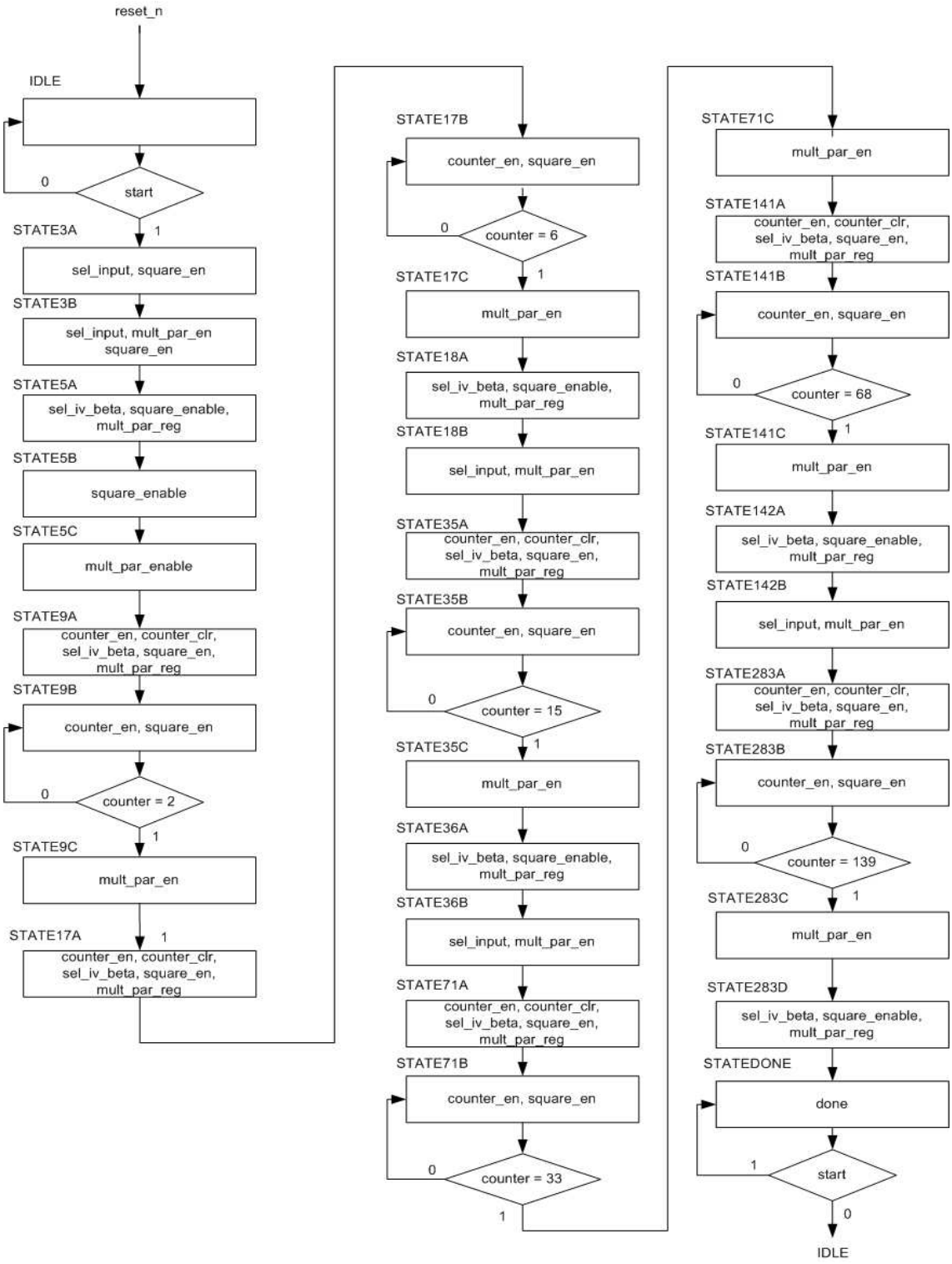

Figure A.18: ASM for the control circuit for the inverter in $GF(2^{283})$ using Fermat's little theorem and based on mult_par

APPENDIX B

Programmer's Model

B.1 Introduction

What follows is a description of the registers to aid programming of the module. The registers addresses are offset from the base address Tatebase.

B.2 Summary of Registers

Address	Type	Width	Reset Value	Name	Description
Tatebase + 0x00	Wr	32	0x00000000	tate_data_in	Data in register
Tatebase + 0x04	Rd	32	0x00000000	tate_data_out	Data out register
Tatebase + 0x08	Wr/Rd	32	0x00000000	tate_control	Control register
Tatebase + 0x0C	Rd	32	0x00000000	tate_status	Status register

Table B.1: Register Summary

B.3 Register Descriptions

tate_data_in [32] +0x00

tate_data_in is the data write register. Data written to this register address is transferred

into the internal registers in the Tate pairing algorithm accelerator block.

Bit	Name	Type	Function
31:0	data	Wr	Write data

Table B.2: tate_data_in

tate_data_out [32] +0x04

tate_data_out is the data read register. Reading from this register is in effect reading the internal output register in the Tate pairing algorithm accelerator block.

Bit	Name	Type	Function
31:0	data	Rd	Read data

Table B.3: tate_data_out

tate_control [32] +0x08

tate_control is the control register. Bit's zero and one control reset the read and write pointer which are used to fill the internal register of the Tate block. Bit two (tate_load) enables the loading of the data into the Tate block and bit three (tate_start) initiates the calculation of the algorithm.

Bit	Name	Type	Function
0	counter_clr_rd	Wr/Rd	reset data read pointer
1	counter_clr_wr	Wr/Rd	reset data write pointer
2	tate_load	Wr/Rd	load data into accelerator
3	tate_start	Wr/Rd	start Tate pairing calculation
31:4		Wr/Rd	not used

Table B.4: tate_control

tate_status [32] +0x0C

tate_status is the status registers and it records the tate_done signal so that it may be read by software. A write, of any data, to this register clears the tate_done bit.

Bit	Name	Type	Function
1	tate_done	Rd	Tate algorithm has completed
31:0		Wr	A write to the status register clears the tate_done bit
31:1		Rd	not used

Table B.5: tate_status

APPENDIX C

Signal Descriptions

C.1 Introduction

The signals that interface with the module are described below. There are two set of signals; the AMBA APB signals and the additional on-chip signals.

C.2 APB Signal

The module is a slave of the AMBA APB. Below are the signals required to connect the device to this bus. All signals are active high.

Name	Type	Source	Description
PCLK	Input	APB	50MHz clock to time bus transfers.
PENABLE	Input	APB	The APB enable signal is asserted high for one clock cycle to enable a bus transfer.
PSEL	Input	APB	When asserted indicates that the module has been selected by the APB bridge and data transfer is requested.
PWRITE	Input	APB	When asserted high this indicates a bus write, when asserted low a bus read is required.
PADDR[7:2]	Input	APB	APB address bus
PWDATA[31:0]	Input	APB	APB write data bus
PRDATA[31:0]	Input	APB	APB read data bus

Table C.1: APB Signal Descriptions

C.3 On-Chip Signals

These are the non APB signals that the module requires or generates.

Name	Type	Source	Destination	Description
tate_clk	Input	Clock Generator	Tate block, Synchroniser block	125MHz system clock
reset_n	Input	Reset Controller	All blocks	Global reset
tate_intr	Output	Tate Block	Interupt Controller	An active high interrupt to indicate that the algorithm is finished

Table C.2: On-Chip Signal Descriptions

BIBLIOGRAPHY

[1] L. M. Adleman, "The function field sieve," in *Algorithmic Number Theory*, ser. Lecture Notes in Computer Science, vol. 877. Springer, 1994, pp. 108–121.

[2] R. Adler, M. Flanigan, J. Huang, R. Kling, N. Kushalnagar, L. Nachman, C.-Y. Wan, and M. Yarvis, "Intel mote 2: an advanced platform for demanding sensor network applications," in *SenSys '05: Proceedings of the 3rd international conference on Embedded networked sensor systems*. New York, NY, USA: ACM Press, 2005, pp. 298–298.

[3] "Sensory Silicon," Akustica. [Online]. Available: http://www.akustica.com

[4] I. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "Wireless sensors networks: a survey," *Computer Networks*, vol. 38, pp. 393–422, Mar. 2002.

[5] "ARM922T," ARM. [Online]. Available: http://www.arm.com

[6] P. S. L. M. Barreto, S. Galbraith, C. O. hEigeartaigh, and M. Scott, "Efficient pairing computation on supersingular abelian varieties," Cryptology ePrint Archive, Report 2004/375, 2004, http://eprint.iacr.org/.

[7] P. S. L. M. Barreto, B. Libert, N. McCullagh, and J.-J. Quisquater, "Efficient and provably-secure identity-based signatures and signcryption from bilinear maps," in *Advances in Cryptology - ASIACRYPT*, ser. Lecture Notes in Computer Science, vol. 3788. Springer, 2005, pp. 515–532.

[8] P. S. L. M. Barreto, B. Lynn, and M. Scott, "Efficient implementation of pairing-based cryptosystems," *J. Cryptol.*, vol. 17, no. 4, pp. 321–334, 2004.

[9] Berkeley. (2006) PicoRadio project. [Online]. Available: http://bwrc.eecs.berkeley.edu/Research/Pico_Radio

[10] M. Bhardwaj, R. Min, and A. Chandrakasan, "Power-aware systems," in *Proc. of 34th Asilomar Conference on Signals, Systems and Computers*, Nov. 2000.

[11] I. Blake, G. Seroussi, N. Smart, and J. W. S. Cassels, *Advances in Elliptic Curve Cryptography (London Mathematical Society Lecture Note Series)*. New York, NY, USA: Cambridge University Press, 2005.

[12] I. F. Blake, G. Seroussi, and N. P. Smart, *Elliptic curves in cryptography*. New York, NY, USA: Cambridge University Press, 1999.

[13] D. Boneh and M. Franklin, "Identity-based encryption from the Weil pairing," *SIAM J. of Computing*, vol. 32, no. 3, pp. 586–614, 2003.

[14] S. D. Brown and Z. G. Vranesic, *Fundamentals of Digital Logic with VHDL Design*. McGraw-Hill Higher Education, 2000.

[15] H. Brunner, A. Curiger, and M. Hofstetter, "On computing multiplicative inverses in GF(2m)," *IEEE Transactions on Computers*, vol. 42, no. 8, pp. 1010–1015, Aug. 1993.

[16] K. M. Butler, "Estimating the economic benefits of DFT," *IEEE Des. Test*, vol. 16, no. 1, pp. 71–79, 1999.

[17] J. C. Cha and J. H. Cheon, "An identity-based signature from gap diffie-hellman groups." in *Public Key Cryptography*, 2003, pp. 18–30.

[18] H. Chan and A. Perrig, "Security and privacy in sensors networks," *IEEE Computer*, vol. 36, no. 10, pp. 103–105, Oct. 2003.

[19] A. Chandrakasan, R. Min, M. Bhardwaj, S.-H. Cho, and A. Wang, "Power aware wireless microsensor systems," in *Proc. ESSCIRC*, Florence,Italy, Sep. 2002, pp. 34–44.

[20] W. Chen, L. Chen, Z. Chen, and S. Tu, "Wits: A wireless sensor network for intelligent transportation system," *imsccs*, vol. 2, pp. 635–641, 2006.

[21] S. Y. Cheung, S. Coleri, B. Dundar, S. Ganesh, C.-W. Tan, and P. Varaiya, "Traffic measurement and vehicle classification with a single magnetic sensor," *Journal of Transportation Research Board*, 2005.

[22] C.-Y. Chong and S. P. Kumar, "Sensor networks: Evolution, opportunities, and challenges," in *Proc. IEEE special issue on Sensor Networks and Applications*, vol. 91, no. 8, Aug. 2003, pp. 1247–1256.

[23] D. Coppersmith, "Fast evaluation of logarithms in fields of characteristic two," *IEEE Transactions on Information Theory*, vol. 30, no. 4, pp. 587–594, Jul. 1984.

[24] *MICA2 Wireless Measurement System*, Crossbow Technology. [Online]. Available: http://www.xbow.com

[25] D. Culler, D. Estrin, and M. Srivastava, "Guest editors' introduction: Overview of sensor networks," *Computer*, vol. 37, no. 8, pp. 41–49, Aug. 2004.

[26] J. R. Douceur, "The sybil attack." in *IPTPS*, 2002, pp. 251–260.

[27] I. M. Duursma and H.-S. Lee, "Tate pairing implementation for hyperelliptic curves $y^2 = x^p - x + d$." in *ASIACRYPT*, 2003, pp. 111–123.

[28] T. ElGamal, "A public key cryptosystem and a signature scheme based on discrete logarithms," *IEEE Trans. Inform. Theory*, vol. 31, no. 4, pp. 469–472, Jul. 1985.

[29] L. Eschenauer and V. D. Gligor, "A key-management scheme for distributed sensor networks," in *Proc. 9th ACM Conference on Computer and Communications Security*, Washington,DC,USA, Nov. 2002, pp. 41–47.

[30] D. F. Finchelstein, B. H. Calhoun, D. C. Daly, N. Verma, D. D. Wentzloff, A. Wang, S.-H. Cho, and A. P. Chandrakasan, "Design considerations for ultra-low energy wireless microsensor nodes," *IEEE Trans. Comput.*, vol. 54, no. 6, pp. 727–740, 2005.

[31] D. F. Finchelstein, S. M.-B. H. Calhoun, S. M.-D. C. Daly, S. M.-N. Verma, S. M.-D. D. Wentzloff, M.-A. Wang, M.-S.-H. Cho, and F.-A. P. Chandrakasan, "Design considerations for ultra-low energy wireless microsensor nodes," *IEEE Trans. Comput.*, vol. 54, no. 6, pp. 727–740, 2005.

[32] K. Fong, D. Hankerson, J. Lopez, and A. Menezes, "Field inversion and point halving revisited," *IEEE Transactions on Computers*, vol. 53, no. 8, pp. 1047–1059, Aug. 2004.

[33] S. D. Galbraith, K. Harrison, and D. Soldera, "Implementing the Tate pairing," in *ANTS-V: Proceedings of the 5th International Symposium on Algorithmic Number Theory*. London, UK: Springer-Verlag, 2002, pp. 324–337.

[34] T. Gao, D. Greenspan, M. Welsh, R. R. Juang, and A. Alm, "Vital signs monitoring and patient tracking over a wireless network," in *Engineering in Medicine and Biology Society, 2005. IEEE-EMBS 2005. 27th Annual International Conference of the*, Sep. 2005, pp. 102–105.

[35] P. Grabher and D. Page, "Hardware acceleration of the tate pairing in characteristic three." in *Cryptographic Hardware and Embedded Systems - CHES 2005*, ser. Lecture Notes in Computer Science, vol. 3659. Springer, pp. 398–411.

[36] R. Granger, D. Page, and M. Stam, "Hardware and software normal basis arithmetic for pairing-based cryptography in characteristic three," *IEEE Transactions on Computers*, vol. 54, no. 7, pp. 852–860, Jul. 2005.

[37] J. Guajardo and C. Paar, "Itoh-Tsujii inversion in standard basis and its application in cryptography and codes," *Des. Codes Cryptography*, vol. 25, no. 2, pp. 207–216, 2002.

[38] V. Gutnik and A. P. Chandrakasan, “Embedded power supply for low-power DSP,” *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 5, no. 4, pp. 425–435, Dec. 1997.

[39] D. Hankerson, A. J. Menezes, and S. Vanstone, *Guide to Elliptic Curve Cryptography*. Secaucus, NJ, USA: Springer-Verlag New York, Inc., 2003.

[40] N. Hashmi, D. Myung, M. Gaynor, and S. Moulton, “A sensor-based, web service-enabled, emergency medical response system,” in *EESR ’05: Proceedings of the 2005 workshop on End-to-end, sense-and-respond systems, applications and services*. Berkeley, CA, USA: USENIX Association, 2005, pp. 25–29.

[41] W. Heinzelman, A. Sinha, A. Wang, and A. Chandrakasan, “Energy-scalable algorithms and protocols for wireless microsensor networks,” in *Proc. International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, Jun. 2000.

[42] A. Hierlemann, “Integrated chemical microsensor systems in CMOS-technology,” in *Solid-State Sensors, Actuators and Microsystems, 2005. Digest of Technical Papers. TRANSDUCERS ’05. The 13th International Conference on*, vol. 2, Jun. 2005, pp. 1134–1137.

[43] J. L. Hill and D. E. Culler, “Mica: a wireless platform for deeply embedded networks,” *IEEE Micro*, vol. 22, no. 6, pp. 12–24, Nov./Dec. 2002.

[44] J. Hill, M. Horton, R. Kling, and L. Krishnamurthy, “The platforms enabling wireless sensor networks,” *Commun. ACM*, vol. 47, no. 6, pp. 41–46, 2004.

[45] J. L. Hill, “System architecture for wireless sensor networks,” Ph.D. dissertation, University of California, Berkeley, 2003. [Online]. Available: http://www.cs.berkeley.edu/~jhill

[46] Y.-C. Hu, A. Perrig, and D. B. Johnson, “Wormhole attacks in wireless networks,” *IEEE Journal on Selected Areas in Communications*, vol. 24, no. 2, pp. 370–380, Feb. 2006.

[47] C. Intanagonwiwat, R. Govindan, and D. Estrin, "Directed diffusion: a scalable and robust communication paradigm for sensor networks," in *MobiCom '00: Proceedings of the 6th annual international conference on Mobile computing and networking*. New York, NY, USA: ACM Press, 2000, pp. 56–67.

[48] "Intel Research," Intel. [Online]. Available: http://www.intel.com/research/exploratory/motes.htm

[49] T. Itoh and S. Tsujii, "A fast algorithm for computing multiplicative inverses in $GF(2^m)$ using normal bases," *Inf. Comput.*, vol. 78, no. 3, pp. 171–177, 1988.

[50] A. Karatsuba and Y. Ofman, "Multiplication of many-digital numbers by automatic computers," *Translation in Physics-Doklady*, vol. 7, pp. 595–596, 1963.

[51] C. Karlof, N. Sastry, and D. Wagner, "TinySec: a link layer security architecture for wireless sensor networks," in *SenSys '04: Proceedings of the 2nd international conference on Embedded networked sensor systems*. New York, NY, USA: ACM Press, 2004, pp. 162–175.

[52] M. Keller, R. Ronan, W. Marnane, and C. Murphy, "A GF(24m) inverter and its application in a reconfigurable tate pairing processor," in *Reconfigurable Computing and FPGA's, 2006. ReConFig 2006. IEEE International Conference on*, San Luis Potosi, Mexico, Sep. 2006, pp. 1–10.

[53] M. Keller, T. Kerins, F. Crowe, and W. Marnane, "Fpga implementation of a $gf(2^m)$ tate pairing architecture," in *International Workshop on Applied Reconfigurable Computing - ARC*, ser. Lecture Notes in Computer Science, vol. 3985. Springer, pp. 358–369.

[54] T. Kerins, W. P. Marnane, E. M. Popovici, and P. S. L. M. Barreto, "Hardware accelerators for pairing based cryptosystems," *IEE Proceedings Information Security*, vol. 152, pp. 47–56, Oct. 2005.

[55] T. Kerins, C. Murphy, C. O hEigeartaigh, R. Ronan, and M. Scott, "FPGA acceleration of the tate pairing in characteristic 2," in *Field Programmable Technology,*

2006. FPT 2006. IEEE International Conference on, Bangkok, Dec. 2006, pp. 213–220.

[56] N. S. Kim, T. Austin, D. Baauw, T. Mudge, K. Flautner, J. S. Hu, M. J. Irwin, M. Kandemir, and V. Narayanan, "Leakage current: Moore's law meets static power," *Computer*, vol. 36, no. 12, pp. 68–75, Dec. 2003.

[57] A. Kinane, "Energy Efficient Hardware Acceleration of Multimedia Processing Tools," Ph.D. dissertation, School of Electronic Engineering, Dublin City University, Apr. 2006. [Online]. Available: http://www.eeng.dcu.ie/~kinanea/thesis/kinane_final.pdf

[58] L. M. Kohnfelder, "Toward a practical public-key ctyptosystem," MIT, Cambridge, Mass., USA, May 1978, bachelor's thesis.

[59] S. Kwon, "Efficient Tate pairing computation for elliptic curves over binary fields." in *ACISP*, 2005, pp. 134–145.

[60] C. H. Lim and H. S. Hwang, "Fast implementation of elliptic curve arithmetic in GF(p^n)." in *Public Key Cryptography, Third International Workshop on Practice and Theory in Public Key Cryptography - PKC*, ser. Lecture Notes in Computer Science, vol. 1751. Springer, 2000, pp. 405–421.

[61] E. D. Mastrovito, "VLSI architectures for computation in Galois fields," Ph.D. dissertation, Linköping University, Linköping, Sweden, 1989.

[62] R. J. McEliece, *Finite fields for computer scientists and engineers*. Kluwer Academic Publishers, 1987.

[63] A. J. Menezes, *Elliptic Curve Public Key Cryptosystems*. London, UK: Kulwer Academic Publishers, 1993.

[64] V. S. Miller, "Short programs for functions on curves," 1986, unpublished. [Online]. Available: http://crypto.stanford.edu/miller/miller.pdf

[65] R. Min and A. Chandrakasan, "A framework for energy-scalable communication in high-density wireless networks," in *ISLPED '02: Proceedings of the 2002 international symposium on Low power electronics and design*. New York, NY, USA: ACM Press, 2002, pp. 36–41.

[66] R. Min, S.-H. Cho, M. Bhardwaj, E. Shih, A. Wang, and A. Chandrakasan, *Power-Aware Design Methodologies*. Kluwer Academic Publishers, 1998, ch. Power-Aware Wireless Microsensor Networks, pp. 335–372.

[67] M.I.T. (2006) μ-adaptive multi-domain power aware sensors. [Online]. Available: http://mtlweb.mit.edu/researchgroups/icsystems/uamps/

[68] G. E. Moore, "Cramming More Components onto Integrated Circuits," *Electronics*, vol. 38, no. 8, Apr. 19 1965.

[69] T. Mudge, "Power: a first-class architectural design constraint," *Computer*, vol. 34, no. 4, pp. 52–58, Apr. 2001.

[70] J. Newsome, E. Shi, D. Song, and A. Perrig, "The sybil attack in sensor networks: analysis & defenses," in *IPSN '04: Proceedings of the third international symposium on Information processing in sensor networks*. New York, NY, USA: ACM Press, 2004, pp. 259–268.

[71] NIST, "Digital signature standard," National Institute of Standard and Technology, Gaithersburg, MD, USA, Tech. Rep. FIPS 186-2, Jan. 2000.

[72] C. Paar, P. Fleischmann, and P. Roelse, "Efficient multiplier architectures for Galois fields $GF(2^{4n})$," *IEEE Trans. Comput.*, vol. 47, no. 2, pp. 162–170, 1998.

[73] M. Perkins, N. Correal, and B. O'Dea, "Emergent wireless sensor network limitations: A plea for advancement in core technologies," Motorola Labs, Plantation, Florida, USA, Tech. Rep., 2002.

[74] A. Perrig, J. Stankovic, and D. Wagner, "Security in wireless sensors networks," *Communications of the ACM*, vol. 47, no. 6, pp. 53–57, Jun. 2004.

[75] J. M. Pollard, “Monte carlo methods for index computation (mod p),” *Math. Comput.*, vol. 32, no. 143, pp. 918–924, 1978.

[76] J. M. Rabaey, M. Ammer, J. L. da Silva Jr., D. Patel, and S. Roundy, “PicoRadio supports ad hoc ultra-low power wireless networking,” *Computer Magazine*, pp. 42–48, Jul. 2000.

[77] J. M. Rabaey, A. Chandrakasan, and B. Nikolic, *Digital Integrated Circuits: A Design Perspective*, 2nd ed. Prentice Hall, 2003.

[78] “AreaRAE IAQ,” RAE Systems. [Online]. Available: http://www.raesystems.com

[79] V. Raghunathan, C. Schurgers, S. Park, and M. B. Srivastava, “Energy-aware wireless microsensor networks,” *IEEE Signal Processing Magazine*, vol. 19, no. 2, pp. 40–50, Mar. 2002.

[80] R. L. Rivest, A. Shamir, and L. M. Adleman, “A method of obaining digital signatures and public key cryptosystems,” *Communications of the ACM*, vol. 21, no. 2, pp. 120–126, Feb. 1978.

[81] R. Ronan, C. O. hEigeartaigh, C. Murphy, M. Scott, T. Kerins, and W. P. Marnane, “An embedded processor for a pairing-based cryptosystem,” in *Information Technology: New Generations, 2006. ITNG 2006. Third International Conference on*, Apr. 2006, pp. 192–197.

[82] S. Roundy, B. Otis, Y. Chee, J. Rabaey, and P. Wright, “A 1.9ghz RF transmit beacon using environmentally scavenged energy,” in *ISLPED*, Seoul, Korea, Aug. 2003.

[83] S. Roundy, D. Steingart, L. Frechette, P. K. Wright, and J. M. Rabaey, “Power sources for wireless sensor networks.” in *EWSN*, 2004, pp. 1–17.

[84] A. Sadat, H. Qu, C. Yu, J. S. Yuan, and H. Xie, “Low-power CMOS wireless MEMS motion sensor for physiological activity monitoring,” *Circuits and Systems*

I: Regular Papers, IEEE Transactions on [see also Circuits and Systems I: Fundamental Theory and Applications, IEEE Transactions on], vol. 52, no. 12, pp. 2539–2551, Dec. 2005.

[85] R. Sakai, K. Ohgishi, and M. Kasahara, "Cryptosystems based on pairing," in *Symposium on Cryptography and Information Security (SCIS2000)*, Okinawa, Japan, Jan. 2000, pp. 26–28.

[86] C. Savarese, "Robust positioning algorithms for distributed ad-hoc wireless sensor networks," Master's thesis, University of California, Berkeley, California, U.S.A., 2002.

[87] C. Savarese, J. M. Rabaey, and J. Beutel, "Locationing in distributed ad-hoc wireless sensor networks," in *Proc. International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, May 2001.

[88] R. C. Shah and J. Rabaey, "Energy aware routing for low energy ad hoc sensor networks," in *Proc. IEEE Wireless Communications and Networking Conference (WCNC)*, Mar. 2002.

[89] A. Shamir, "Identity-based cryptosystems and signature schemes," in *Proc. Crypto '84*, Santa Barbara, California, USA, Aug. 1984, pp. 47–54.

[90] "Multiprecision Integer and Rational Arithmetic C/C++ Library (MIRACL)," Shamus Software. [Online]. Available: http://www.shamus.ie

[91] S. C. Shantz, "From euclid's gcd to montgomery multiplication to the great divide," Mountain View, CA, USA, Tech. Rep., 2001.

[92] E. Shih, S.-H. Cho, N. Ickes, R. Min, A. Sinha, A. Wang, and A. Chandrakasan, "Physical layer driven algorithm and protocol design for energy-efficient wireless sensor networks," in *MobiCom '01: Proceedings of the 7th annual international conference on Mobile computing and networking*. New York, NY, USA: ACM Press, 2001, pp. 272–286.

[93] C. Shu, K. Gaj, and S. Kwon, "FPGA accelerated tate pairing based cryptosystems over binary fields," in *Field Programmable Technology, 2006. FPT 2006. IEEE International Conference on*, Bangkok, Dec. 2006, pp. 173–180.

[94] A. Sinha and A. Chandrakasan, "Energy aware software," in *Proc. of the 13th International Conference on VLSI Design*, Jan. 2000, pp. 50–55.

[95] ——, "Operating system and algorithmic techniques for energy scalable wireless sensor networks," in *MDM '01: Proceedings of the Second International Conference on Mobile Data Management*. London, UK: Springer-Verlag, 2001, pp. 199–209.

[96] A. Sinha, A. Wang, and A. P. Chandrakasan, "Algorithmic transforms for efficient energy scalable computation," in *ISLPED '00: Proceedings of the 2000 international symposium on Low power electronics and design*. New York, NY, USA: ACM Press, 2000, pp. 31–36.

[97] G. Stitt, F. Vahid, and S. Nematbakhsh, "Energy savings and speedups from partitioning critical software loops to hardware in embedded systems," *Trans. on Embedded Computing Sys.*, vol. 3, no. 1, pp. 218–232, 2004.

[98] *TSMC 65nm Technology Platform*, Taiwan Semiconductor Manufacturing Company. [Online]. Available: http://www.tsmc.com

[99] A. Wang and A. Chandrakasan, "Energy efficient system partitioning for distributed wireless sensor networks," in *Proc. International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, May 2001.

[100] A. Wang, W. B. Heinzelman, A. Sinha, and A. P. Chandrakasan, "Energy-scalable protocols for battery-operated microsensor networks," *J. VLSI Signal Process. Syst.*, vol. 29, no. 3, pp. 223–237, 2001.

[101] L. C. Washington, *Elliptic Curves, Number Theory and Cryptography*. London, UK: Chapman & Hall/CRC, 2003.

[102] L. Wei, Z. Chen, K. Roy, M. C. Johnson, Y. Ye, and V. K. De, "Design and optimization of dual-threshold circuits for low-voltage low-power applications," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 7, no. 1, pp. 16–24, Mar. 1999.

[103] S. B. Wicker, *Error control systems for digital communication and storage*. Prentice-Hall, 1995.

[104] Y. Zhang, W. Liu, W. Lou, and Y. Fang, "Location-based compromise-tolerant security mechanisms for wireless sensor networks," *IEEE Journal on Selected Areas in Communications*, vol. 24, no. 2, pp. 247–260, Feb. 2006.

[105] Y. Zheng, "Digital signcryption or how to achieve cost(signature & encryption) $<<$ cost(signature) + cost(encryption)," in *Advances in Cryptology - CRYPTO '97*, ser. Lecture Notes in Computer Science, vol. 1294. Springer, 1997, pp. 165–179.