1-1-2007

Realization of a cryptographic algorithm in FPGA based on an authentication protocol for RFID smart tags

Shirley Arnold
Ryerson University

Realization of a Cryptographic Algorithm in FPGA based on an Authentication Protocol for RFID Smart Tags

By

Shirley Arnold

A project
presented to Ryerson University
in partial fulfillment of the
requirement for the degree of
Master of Engineering
in the Program of
Electrical and Computer Engineering

Toronto, Ontario, Canada, 2007

Author's Declaration

I hereby declare that I am the sole author of this project.

I authorize Ryerson University to lend this project to other institutions or individuals for the purpose of scholarly research.

Signature

I further authorize Ryerson University to reproduce this project by photocopying or by other means, in total or in part, at the request of other institutions or individuals for the purpose of scholarly research.

Signature

Instructions on Borrowers

Ryerson University requires the signatures of all persons using or photocopying this project. Please sign below, and give address and date.

Abstract

Realization of a Cryptographic Algorithm in FPGA based on an Authentication Protocol for RFID Smart Tags

Master of Engineering
Electrical and Computer Engineering
Ryerson University

Radio Frequency Identification (RFID) is expected to become pervasive and ubiquitous, as it can be embedded into everyday items as smart labels. A typical scenario of exploiting RFID is Electronic container seal (E-seal) in container security. Since RFID systems are proposed to be used in such high security applications, cryptographic authentication is necessary to protect the privacy and security of the RFID system itself. In this project, AES128 cryptographic algorithm based on the new approach of authentication protocol was implemented on FPGA environment. A two-way challenge-response authentication scheme is used due to the limited computing power, low die-size, and low-power requirements. The hardware implementation of the algorithm on FPGA provides acceleration and fast prototyping. The positive results in low silicon area ascertain that the authentication algorithm can be implemented in a small RFID tag for more secure system.

Acknowledgments

I would like to greatly acknowledge the supervision and guidance of my supervisor Dr.Xavier Fernando during this Project. I am particularly thankful to him for providing me with exceptional supervision in defining and directing this project work. I would like to thank him for his encouragement and inspiration. I am also much indebted to his patience and willingness through out my work on this project.

I express my deep gratitude to Dr. Gul Khan for providing me with the guidance and resources for my project. His constructive criticism has greatly helped to sharpen the Analysis and clarity of my work. His guidance, relevant material, and suggestions for Altera Stratix hardware development board were helpful in improving this work.

Hearty thanks to all my friends for their diligent and timely help all through my Master's program. I would like to thank one and all in Electrical and Computer Engineering Dept. in Ryerson University who supported me in pursuing my Master's program.

Finally, I would like to express my greatest gratitude to my family. I feel proud and consider myself to be fortunate to have their encouragement and support. I dedicate this work to my lovely daughter Catherine, my husband and my parents.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Radio Frequency Identification (RFID) is an emerging technology. The main idea behind it is to attach a so called RFID tag to every object in a particular environment and give a digital identity to all these objects. An RFID tag is a small microchip, with an antenna, holding a unique ID and other information which can be sent over radio frequency. The information can be automatically read and registered by RFID readers. The data received by the RFID reader can be subsequently processed by a back-end database. RFID technology enables tag reading from a greater distance, even in harsh environments. The readers are used to query RFID tags to obtain identification, location, and other information about the device or product the tag is embedded in. The core of the RFID system consists of RFID tags, RFID readers, and host computer and application software.

1.1 RFID System

The purpose of an RFID system is to enable data to be transmitted by a portable device, called a tag, which is read by an RFID reader and processed according to the needs of a particular application. The data transmitted by the tag may provide identification or location information, or specifics about the product tagged, such as price, color, date of purchase, etc. In a typical RFID system, individual objects are equipped with a small, inexpensive tag. The tag contains a transponder with a digital memory chip that is given a unique electronic product code. The interrogator, an antenna packaged with a transceiver and decoder, emits a signal activating the RFID tag so it can read and write data to it. When an RFID tag passes through the electromagnetic zone, it detects the reader's activation signal. The reader decodes the data encoded in the tag's integrated circuit (silicon chip) and the data is passed to the host computer for processing.

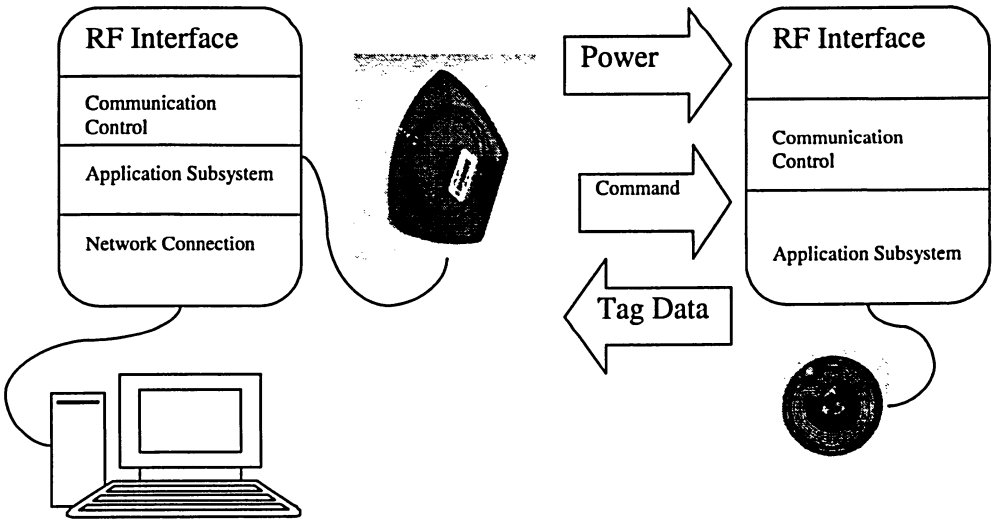


Figure 1.1: RFID System [1]

1.1.1 RFID Tags

RFID tags can be either **passive or active. Passive RFID** tags do not have their own power supply: the minute electrical current induced in the antenna by the incoming radio-frequency scan provides enough power for the tag to send a response. Due to power and cost concerns, the response of a passive RFID tag is brief: typically just an ID number. Passive tags have practical read ranges that vary from about 10 mm up to about 6 meters.

Active RFID tags, on the other hand, must have a power source, and may have longer ranges and larger memories than passive tags, as well as the ability to store additional information sent by the transceiver. Many active tags have practical ranges of tens of meters, and a battery life of up to several years. As passive tags are much cheaper to manufacture and do not depend on a battery, the vast majority of RFID tags in existence are of the passive variety. While the cost advantages of passive tags over active tags are significant, other factors including accuracy and reliability make the use of active tags very common today.

There are four different kinds of tags commonly in use. They are categorized by their radio frequency: low frequency tags (between 125 to 134 KHz), high frequency tags

(13.56 MHz), UHF tags (868 to 956 MHz), and microwave tags (2.45 GHz). UHF tags cannot be used globally as there aren't any global regulations for their usage.

- Low-frequency RFID tags are commonly used for animal identification, and automobile key-and-lock, anti-theft systems. Pets may be embedded with small chips so that they may be returned to their owners if lost.
- High-frequency RFID tags are used in library book or bookstore tracking, pallet tracking, building access control, airline baggage tracking, and apparel item tracking.
- UHF RFID tags are commonly used commercially in pallet and container tracking, and truck and trailer tracking in shipping yards.
- Microwave RFID tags are used in long-range access control for vehicles.

The most popular frequencies are LF (125 KHz), HF (13.56 MHz), UHF 915 MHz, and 2.4 GHz. The original tags used the lower frequencies but the most popular is 13.56 MHz. The UHF frequencies of 915 MHz is growing in use because it provides a longer reading range and is the key operating frequency. To encode information, most RFID systems employ simple AM modulation, but others employ additional layer of modulation on a sub carrier frequency in the form of FSK or BPSK. Backscatter modulation, the common modulation scheme is a form of ASK is the popular modulation scheme. The result is an AM wave. Speeds from a few kilobits per second to several hundred bits per second are typical. The higher the operating frequency, the higher the potential data rate.

1.1.2 RFID Reader

RFID readers are also called interrogators. The fundamental function of a tag reader is to transmit energy to a tag, to detect the response of the tag and to translate the resulting analog signal into meaningful digital data. The reader's main functions are therefore to activate the data carrier (transponder) structure the communication sequence with the data carrier and transfer data between application software and a contact less data carrier. i.e.,

making the connection and performing anti-collision and authentication procedures are handled entirely by the reader. There are two types of RFID readers:

- **RFID read-only readers**: These devices can only query or read information from a nearby RFID tag. They can be found in fixed, stationery applications as well as portable, handheld varieties.
- **RFID read-write readers**: These devices are also known as encoders, they can not only read information, but can also write (change) information in an RFID tag. Such RFID encoders can be used to program information into a "blank" RFID tag.

Readers in all systems can be reduced to two fundamental functional blocks: the control system and RF interface consisting of a transmitter and receiver.

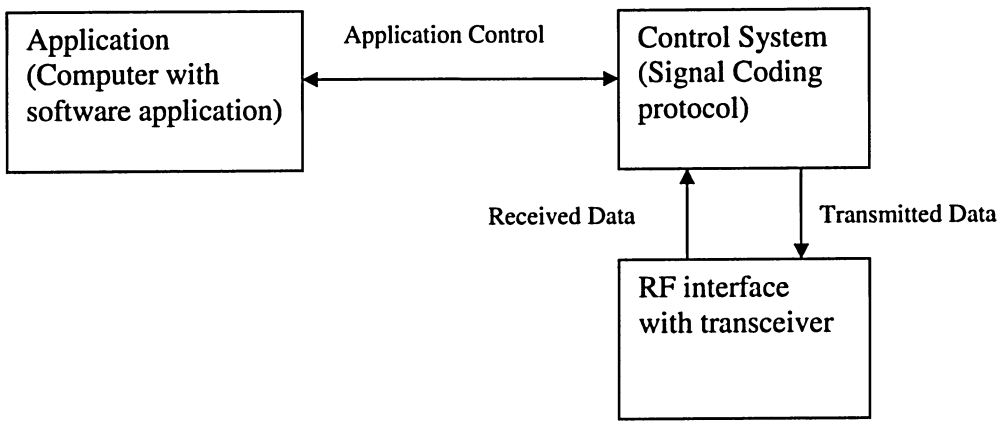


Figure 1.2: Block Diagram of RFID Reader

The reader's RF interface performs the following functions.

- Generation of high frequency transmission power to activate the transponder and supply it with power.
- Modulation of the transmission signal to send data to the transponder.

The reader's control unit performs the following functions:

- Communication with the application software and the execution of commands from the application software.
- Control of the communication with a transponder.
- Signal coding and decoding

In more complex systems the following additional functions available:

- Execution of an anti-collision algorithm.
- Encryption and decryption of the data to be transferred between transponder and reader.
- Performance of authentication between transponder and reader.

The control unit is usually based upon a microprocessor to perform these complex functions crypto logical procedures, such as stream ciphering between transponder and reader and also signal coding is often performed in an additional ASIC module to relieve the processor of calculation intensive processes. Various, often self defined, protocols are used for the communication protocol.

1.2 Applications

RFID technology can be applied to wide range applications from inventory control, Access control to Health Care. The following are some examples of RFID applications that are in currently in use and can be developed in future.

- **Tracking Inventory in warehouse or Inventory control:** Inventory can be updated in real time without product movement, scanning or human involvement. Fully automated systems, which allow inventory status to be determined and shipping & receiving documents can be generated automatically. The systems that can also trigger automatic orders for products those are low in inventory.
- **Container /Pallet Tracking:** Can be programmed to track location of pallets and containers within the warehouse and notify management and security when unscheduled movements occur, which also gives theft control. The cargo containers can be sealed using RFID E-seals.
- **Parking lot Access Control:** can provide businesses and communities with hands-free control to ensure only authorized vehicles have entry. It can also provide access data for administering periodic access charges or parking fees. The

RFID reader can also trigger surveillance cameras or video recorders whenever a vehicle enters or exits the controlled area. [5]

• **Manufacturing Lines:** Manufacturers can track and record in-process assembly information into the RFID tag as an item progresses along the line. This is ideal for manufacturers who build several products on a single production line, or manufacture complex or customized products.

• **ID Badges and Access Control:** RFID technology provides a hands-free access control solution with many advantages over traditional access control badges and systems. Badges can be read up to 5 meters (~16 feet) from the access control doorway or portal. This is important for handicapped workers. [5]

• **Airports and High Security:** RFID access control and personnel tracking and location systems can help to assure the security of restricted areas in airports, such as flight lines, baggage handling areas, customs, employee lounges, and other sensitive areas. RFID systems can be used to track employee and passengers in real time.

RFID FOR HEALTH CARE

Since RFID can be engineered to work with no special user action (i.e. passive detection), it finds number of applications in health care.

• **Better elder care –** Augment a medicine cabinet to know when to take pills and how many and the pill expiration date. [6]

Figure 1.3 Medicine Cabinet with RFID reader and Tagged Medicine

• **ADL Monitoring** – Activities of daily living monitoring can be done with RFID technology for activity inferencing of elderly people and people with special needs. This can also be improved as **Healthy Smart Home** for Disabled and elderly people. [6]

• **Infant Care** – Sensors for autism caregivers and could use RFID to augment the child behavior observation system (CBOS). This can be done by having a toddler wear a small portable RFID reader and it broadcasts "toddler nearby". Nearby RFID tags listen to that message, consult local code snippets ("taglets") and decide what to do

- Lock the Drano cap
- Lock the kitchen cabinet
- Turn off the oven burner
- Unlock the fridge door
- Track the toys

• **Blood Transfusion safety** – RFID solutions to increase the efficiency and reliability of blood transfusion safety in operating rooms to accurately identify bags of blood used for transfusions. [6]

The range of possible applications varies with the capability of the tag and is separated by different classes. Class 0 (Passive identity tag) and Class 1 (Passive functional tag) RFID tags are used as barcode replacement and are read-only or can be programmed only once in the field, respectively. Inventory maintenance which is used in the supply chain management can be automated using such tags. They are cheap (approximately 5 Cents) and can be used on item-level on nearly every product.

1.3 Motivation

The applications for more advanced RFID systems are manifold but especially high-value products like pharmaceutical and branded goods can be protected against security vulnerabilities.

The basic functionality of RFID systems is to provide identification of individual objects by the replies the attached RFID tag sends to a request performed by a reader. The reader uses an attached database to link the received ID number to a specific object described in the database. The major drawback of those systems is that the

communication scheme does not provide a method to prove the claimed identity. Since a typical tag answers its ID to any reader (without a possibility to check whether a reader is authorized to receive the information), and the replied ID is always the same, an attacker can easily forge the system by reading out the data of a tag and duplicating it to bogus tags. Closed RFID systems with common access of all readers to a central database, can check for illegal duplicates (bogus tags) within the database but this is not practical for many applications. Furthermore, it is impossible to distinguish the original tag from its illegal duplicates.

Strong authentication mechanisms can solve uprising security problems in RFID systems and therefore give protected tags an added value. The three main security threats in RFID systems are forgery of tags, unwanted tracking of customers, and the unauthorized access to the tag's memory. Martin Feldhofer and Manfred Aigner have proposed authentication protocols for RFID systems based on the ISO/IEC 9798-2 standard [7]. These protocols allow protecting high-value goods against adversary attackers. And, they are feasible for restriction concerning data rates and compliance to existing standards as well as the requirements concerning chip area and power consumption. Authentication is a method to provide a proof for a claimed identity. This proof is based on a secret stored within the authenticating part of the system. As long as the secret information stays secret and the used protocol does not leak sensitive information, an attacker cannot forge a tag.

A communication system providing authentication can reject access (to information, entry, etc.) to non authorized parties. To keep the authentication secure, it is necessary that an attacker does not gain information about the secret by listening passively to successful authentications. To fulfill this requirement for strong authentication, it is necessary to use cryptographically strong computations.

"Cryptographically strong" in this context means that it must be computationally infeasible with current computing systems to derive the secret key data from an unlimited number of known input and output message pairs.

1.4 Objective and Organization

This project is focusing on more advanced tags Class 2(Semi-Passive Tags) which also have a rewritable memory and additional hardware resources with and without an active power supply on the tag. For the tags without active power supply, the energy for operation is pulled from the electromagnetic field provided by the reader. In addition, the reader also provides the digital clock frequency for operation. Such tags cost about 50 cents and the available silicon area is about 10,000 gates.

The objective of this work is to implement AES128 cryptographic algorithm in FPGA and test for signals from RFID receiver in order to calculate the silicon area, delay and power consumption. This study would help in optimization of implementation of cryptographic algorithm in a tag with minimal power supply. The thesis is organized into five chapters: Chapter 2 discusses the necessary background on privacy and security issues of the RFID systems and approaches proposed for privacy protection and integrity assurance. Chapter 3 provides theoretical details on AES cryptographic algorithm and FPGA implementation of the algorithm. Chapter 4 deals with the results of the implementation. Chapter 5 summarizes the work and suggests possibilities for future work.

Chapter 2

Survey on RFID Privacy and Security

In both the popular press and academic circles, RFID has seen a swirl of attention in the past few years. Developed as a means of tracking the movement of products, and even people, the chips have been at the centre of some heated arguments. Privacy campaigners paint frightening pictures of people being tracked by a central computer. The technology's supporters offer an idyllic image of a future where dangerous criminals can be tagged and you could breeze through supermarkets and have the bill paid automatically by your debit card simply by walking past a scanner at the door without having to queue.

Many of us already use RFID tags routinely. Examples include Proximity cards, that is, the contactless cards used for building access, automated toll-payment transponders – the small plaques mounted in automobile windshields. (These are usually semi-passive), the ignition keys of many millions of automobiles, which include RFID tags as a theft-deterrent and Payment tokens like the SpeedPass™ token for petrol station payments. But there is more controversial use of RFID like 'chipping of normal members of the public using a tag called the VeriChip™, a device no larger than a grain of rice. Controversy abounds with RFID and there are concerns about how secure and private the technology is. This chapter discuss on RFID security and privacy problems and possible solutions in literature.

2.1 Security and Privacy Problems

2.1.1 Privacy

RFID raises two main privacy concerns for users: clandestine *tracking* and *inventorying*. RFID tags respond to reader interrogation without alerting their owners or bearers. Thus, where read range permits, clandestine scanning of tags is a plausible threat. In consequence, a person carrying an RFID tag effectively broadcasts a fixed serial number to nearby readers, providing a ready vehicle for clandestine physical tracking. Such tracking is possible even if a fixed tag serial number is random and carries no intrinsic data. [3]

The threat to privacy grows when a tag serial number is combined with personal information. For example, when a consumer makes a purchase with a credit card, a shop can establish a link between her identity and the serial numbers of the tags on her person. Marketers can then identify and profile the consumer using networks of RFID readers, both inside shops and without. The problem of clandestine tracking is not unique to RFID, of course. It affects many other wireless devices, such as Bluetooth-enabled ones. [13]. In addition to their unique serial numbers, certain tags such as EPC tags in particular carry information about the items to which they are attached. EPC tags include a field for the "General Manager," typically the manufacturer of the object, and an "Object Class," typically a product code, known formally as a Stock Keeping Unit [4]. Thus a person carrying EPC tags is subject to clandestine inventorying. A reader can silently determine what objects she has on her person, and harvest important personal information: What types of medications she is carrying, and therefore what illnesses she may suffer from; the RFID enabled loyalty cards she carries, and therefore where she shops; her clothing sizes and accessory preferences, and so forth. This problem of inventorying is largely particular to RFID. Today the problems of clandestine RFID tracking and inventorying are of limited concern, since RFID infrastructure is scarce and fragmentary. The tagging of individual retail items is probably some years away. Once RFID becomes pervasive, however, as is almost inevitable, the privacy problem will assume more formidable dimensions.

Tag read ranges are an important factor in discussions about privacy. Different operating frequencies for tags induce different ranges, thanks to their distinctive physical properties. Under ideal conditions, for instance, UHF tags have read ranges of over ten meters; for HF tags, the maximum effective read distance is just a couple of meters. Additionally, environmental conditions impact RFID efficacy. The proximity of radio-reflective materials, e.g., metals, and radio-absorbing materials, like liquids, as well as ambient radio noise, affect scanning distances. Protocol and hardware-design choices also affect read ranges. The importance of RFID privacy in military operations reinforces an oft-neglected point: Privacy is not just a consumer concern. The enhanced supply-chain visibility that makes RFID so attractive to industry can also, in another guise, betray competitive intelligence. Enemy forces monitoring or harvesting RFID communications in a military supply chain could learn about troop movements. In civilian applications, similar risks apply. For example, many retailers see item-level RFID tagging as a means to monitor stock levels on retail shelves, and avoid out-of-stock products. Individually tagged objects could also make it easier for competitors to learn about stock turnover rates; corporate spies could walk through shops surreptitiously scanning items. It is useful to bear in mind the full scope of the privacy problem. For example a store's inventory labeled with unprotected tags may be monitored by competitors' unauthorized readers. The inventory data holds significant financial value for commercial organization and competitors.

2.1.2 Security

Privacy in media coverage of RFID to some extent has overshadowed the equally significant problem of authentication. RFID privacy concerns the problem of misbehaving readers harvesting information from well-behaving tags. RFID authentication, on the other hand, concerns the problem of well behaving readers harvesting information from misbehaving tags, particularly counterfeit ones. Basic RFID tags are vulnerable to simple counterfeiting attacks. Scanning and replicating such tags requires little money or expertise. In [14], Jonathan Westhues, an undergraduate student, describes how he constructed what is effectively an RF tape-recorder. This device can read commercial proximity cards – even through walls – and simulate their signals to compromise building entry systems.

EPC tags will be vulnerable to similar attacks. An EPC, after all, is just a bit string, copyable like any other. EPC tags offer no real access-control mechanisms. It is possible that "blank," i.e., fully field-programmable EPC tags, will be readily available on the market. [18] More importantly, elementary RFID simulation devices will be easy to come by or create. Such devices need not even resemble RFID tags in order to deceive RFID readers. As a result, EPC tags may carry no real guarantee of authenticity. Yet plans are afoot for use of such tags as anti-counterfeiting devices. In the United States, the Food and Drug Administration (FDA) has called for the pharmaceutical industry to apply RFID tags to pallets and cases by 2007, with the aim of combating counterfeit pharmaceuticals [16]. Two companies, Texas Instruments and VeriSign Inc., have proposed a "chain of- custody" approach in support of this effort [17]. Their model involves digital signing of tag data to provide integrity assurance. Digital signatures do not confer cloning resistance to tags, however. They prevent forging of data, but not copying of data. Researchers at Johns Hopkins University and RSA Laboratories recently identified a serious security weakness in the RFID tag in Speedpass devices and many automobile immobilizer systems [26]. By demonstrating that such tags could be cloned, the researchers revealed the possibility of payment fraud and new modes of automobile theft. Although their discovery doesn't directly undermine consumer privacy, it demonstrates that RFID tags could have security consequences beyond merely tracking or profiling consumers [27].

Even in the absence of resistance to tag cloning, unique numbering of objects can be a powerful anti-counterfeiting tool. RFID tags can help combat counterfeiting on the principle that if two RFIDtagged crates turn up in a warehouse with identical serial numbers, a problem has clearly arisen. Such detection does not require tag authentication. The FDA has noted that simply by furnishing better data on item pedigrees in supply chains, RFID tags can help identify sources of counterfeit goods. But there are scenarios in which counterfeiters can exploit the vulnerability of RFID tags to cloning. Detection of duplicates ultimately require consistent and centralized data collection; where this is lacking, physical and digital anticounterfeiting mechanisms become more important. Some RFID devices, such as the American Express ExpressPay™ and the MasterCard PayPass™ credit cards, and the active RFID tags that will secure shipping containers, can

perform cryptographic operations. Except for reverse-engineering, these devices offer very good resistance to cloning. There is another aspect of authentication that is specific to RFID, namely authentication of *distance*. Thanks to the relatively short range of some RFID devices, users can authorize commercial transactions with RFID devices by placing them explicitly in proximity to readers. RFID-enabled payment tokens like credit cards work this way. However, tag distance is difficult to authenticate. Researchers have already demonstrated spoofing attacks. The problems of RFID security and privacy are to some extent interdependent. Generally, cloning a tag requires scanning that violates the privacy of its holder.

2.1.3 Attack models

In most cryptographic models, the adversary is assumed to have more-or-less unfettered access to system components in the runtime environment. In security models for the Internet, this makes sense: An adversary can more or less access any networked computing device at any time. A server, for instance, is always on-line, and responds freely to queries from around the world. For RFID systems, however, around the-clock access by adversaries to tags is usually too strong an assumption. In order to scan a tag, an adversary must have physical proximity to it – a sporadic event in most environments. Many cryptographic models of security fail to express important features of RFID systems. A simple cryptographic model, for example, captures the top-layer communication protocol between a tag and reader. At the lower layers are anti-collision protocols and other basic RF protocols. Avoine and Oechslin [19] importantly enumerate the security issues present at multiple communication layers in RFID systems. Among other issues, they highlight the risks of inadequate random-number generation in RFID tags. There is, however, a flip side to the presence of multiple communication layers in tags. If tags have distinct radio fingerprints that are sufficiently difficult to reproduce in convincing form factors, then these fingerprints could help strengthen device authentication.

2.2 Proposed Remedies

Basic RFID tags, lack the resources to perform true cryptographic operations like encryption, strong pseudorandom number generation, and hashing. Low-cost tags, such as EPC tags, possess at most a couple of thousand gates, devoted mainly to basic operations [20]. Few gates – on the order of hundreds – remain for security functionality. Thus, given the choice between, say, a ten-cent RFID tag that can do cryptography, and a five cent tag that cannot, it seems inevitable that most retailers and manufacturers will plump for the five-cent tag. They will address security and privacy concerns using other, cheaper measures. The various proposed approaches for these basic tags is discussed below.

2.2.1 Killing and Sleeping

The designers realized that EPC tags might be irretrievably embedded in consumer devices and that consumer might not want to be tracked. They viewed killing EPC tags at the point-of-sale as an easy way out of the apparent privacy dilemma. The underlying principle is that "dead tags don't talk." When an EPC tag receives a "kill" command from a reader, it renders itself permanently inoperative. To prevent wanton deactivation of tags, this kill command is PIN protected. To kill a tag, a reader must also transmit a tag-specific PIN. As an alternative to killing, tags can also be attached to a product's price tag and discarded at the point-of-sale. Killing or discarding tags enforces consumer privacy effectively, but it eliminates all of the post-purchase benefits of RFID for the consumer [21].

In addition to facilitating item returns or repairs, there are numerous other possible practical uses for RFID tags. People who are physically or mentally impaired might benefit from RFID-based home aids that use tag information. For example; a personal RFID reader could help blind people by reading labels to reveal product contents. And in some cases, such as libraries and rental shops, RFID tags cannot be killed because they must survive over the lifetime of the objects they track. For these reasons, it is imperative to look beyond killing for more balanced approaches to

consumer privacy. Rather than killing tags at the point of sale, then, why not put them to "sleep," i.e., render them only temporarily inactive? This concept is simple, but would be difficult to manage in practice. Clearly, sleeping tags would confer no real privacy protection if any reader at all could "wake" them. Therefore, some form of access control would be needed for the waking of tags. This access control might take the form of tag specific PINs, much like those used for tag killing. To wake a sleeping tag, a reader could transmit this PIN. The sticking point in such a system is that the consumer would have to manage the PINs for her tags. Tags could bear their PINs in printed form, but then the consumer would need to key in or optically scan PINs in order to use them. PINs could be transmitted to the mobile phones or smartcards of consumers – or even over the Internet to their home PCs. Consumers have enough difficulty just managing passwords today, however.

2.2.2 Re-labeling

Sarma, Weis, and Engels propose the idea of effacing unique identifiers in tags at the point of sale [22] to address the tracking problem, but retaining product-type identifiers (traditional barcode data) for later use. As a physical mechanism for realizing the idea they also explore the idea of splitting product-type identifiers and unique identifiers across *two* RFID tags. By peeling off one of these two tags, a consumer can reduce the granularity of tag data. As a remedy for clandestine scanning of library books, Good et al. [23] propose the idea of relabeling RFID tags with random identifiers on checkout. The limitations of these approaches are clear. Effacement of unique identifiers does not eliminate the threat of clandestine inventorying. Nor does it quite eliminate the threat of tracking. Even if tags emit only product-type information, they may still be uniquely identifiable in constellations, i.e., fixed groups. Use of random identifiers in place of product codes addresses the problem of inventorying, but does not address the problem of tracking. To prevent tracking, identifiers must be refreshed on a frequent basis.

While high-powered devices like readers can relabel tags for privacy, tags can alternatively relabel themselves. Juels in [24] proposes a "minimalist" system in which every tag contains a small collection of pseudonyms; it rotates these pseudonyms, releasing a different one on each reader query. An authorized reader can store the full

pseudonym set for a tag in advance, and therefore identify the tag consistently. An unauthorized reader, however, that is, one without knowledge of the full pseudonym set for a tag, is unable to correlate different appearances of the same tag. To protect against an adversarial reader harvesting all pseudonyms through rapid-fire interrogation, Juels proposes that tags "throttle" their data emissions, i.e., slow their responses when queried too quickly. As an enhancement to the basic system, valid readers can refresh tag pseudonyms. The minimalist scheme can offer some resistance to corporate espionage, like clandestine scanning of product stocks in retail environments.

2.2.3 The Proxying Approach

Rather than relying on public RFID readers to enforce privacy protection, consumers might instead carry their own privacy-enforcing devices for RFID. As already noted, some mobile phones include RFID functionality. They might ultimately support privacy protection. Researchers have proposed several systems along these lines like "Watchdog Tag," essentially an audit system for RFID privacy. The watchdog Tag monitors ambient scanning of RFID tags, and collects information from readers, like their privacy policies. And "RFID Guardian" acts as a find of personal RFID firewall. It intermediates reader requests to tags; viewed another way, the Guardian selectively simulates tags under its control. As a high powered device with substantive computing power, a Guardian can implement sophisticated privacy policies, and can use channels other than RFID to supplement ambient data.

2.2.4 Blocking

According to Juels, Rivest, and Szydlo Blocking is a privacy-protecting scheme which depends on the incorporation into tags of a modifiable bit called a *privacy bit.* A '0' privacy bit marks a tag as subject to unrestricted public scanning; a '1' bit marks a tag as "private." JRS refer to the space of identifiers with leading '1' bits as a *privacy zone.* A *blocker tag* is a special RFID tag that prevents unwanted scanning of tags mapped into the privacy zone.

The RFID blocker tag takes a different approach to enhancing RFID privacy. It involves no modification to consumer tags. Rather, the blocker tag creates an RF

environment that is hostile to RFID readers. The blocker tag is a specially configured, ancillary RFID tag that prevents unauthorized scanning of consumer items. In a nutshell, the blocker tag "spams" misbehaving readers so they can't locate the protected tags' identifiers. At the same time, it permits authorized scanners to proceed normally. If two RFID tags simultaneously transmit their identifiers to a reader, a broadcast collision occurs that prevents the reader from deciphering either response. To avoid this problem, RFID readers and tags engage in *singulation*, an anti-collision protocol. An example is the *tree-walking* protocol, in which k-bit identifiers are viewed as binary tree leaves of depth k. In this tree, a node represents a binary identifier prefix. Its left child represents the prefix with a 0 appended; the right child, the prefix with a 1 appended. For a given tree node at depth I (representing an i-bit identifier prefix), the reader starts at the tree's root ($i = 0$) and asks all subtree tags to broadcast their next bit—that is, $(i + 1)$ st. If all tags broadcast a 0, then the reader recurses on the left subtree; if all tags broadcast a 1, then the reader recurses on the right. If some tags broadcast a 0 and some broadcast a 1, then the reader recurses on both subtrees. The blocker tag spoofs the tree-walking protocol into thinking that all tags—that is, all identifiers—are present. To do this, it simply emits both a 0 and a 1 in response to all reader queries. The result is that the reader attempts to traverse the entire identifier tree, believing that all possible tag identifiers in the world are present! The reader stalls because the tree is far too big to be fully scanned (for Class 1 EPC tags, the tree would have 296 nodes). It's likewise easy to construct a blocker tag that will disrupt an EPC reader. An EPC tag that doesn't totally disrupt all RFID activity requires a few refinements, however. The first such refinement is to designate a *privacy zone.* This is a portion of the tree that the blocker simulates—says, the tree's right half, where all identifiers begin with a 1 bit. Tags to be protected by the blocker should then carry a leading 1 bit, whereas freely scannable tags carry a 0 bit. As a simple example, supermarket items might all carry tags with leading 0 bits, which would be flipped at the checkout register (when an appropriate PIN is provided). Supermarket bags could carry blocker tags, protecting the items from scanning until the consumer takes them home and removes them from the bags. At that point, if the items were placed in a "smart" refrigerator or similar device, their tags could again be scanned.

With a *polite blocking* enhancement, the blocker tag would inform readers that it's present, so that they don't attempt to scan the privacy zone and subsequently stall.

A sophisticated adversary might well be able to design or configure a reader that sometimes defeats blocker tags. Blocker tags aim to enhance consumer privacy and make privacy violations more difficult, but they certainly provide nothing like foolproof protection. Also, impolite or even malicious blockers impose a denial-of-service threat. This threat is always present, however, and is not a good justification for refraining from polite blocker-tag deployment. *Soft blocking* is an alternative, lightweight approach to blocker-tag deployment [28]. The basic idea is to enforce polite reader behavior by ensuring that they always adhere to a "blocker-compliant or "polite" policy. We might accomplish this by requiring that polite reader firmware be the commercial default, as well as using auditing procedures and legislative regulation. If readers adhere to a polite policy, a blocker tag can confer privacy protection merely by informing a reader of its presence. Thus, in the soft blocking approach, blocker tags might be ordinary RFID tags whose only special distinction is a special tag identifier, say, a *B* for blockers. This would make blocker tags especially easy to manufacture. In soft blocking, a polite reader would begin a scan by checking for tags with the initial serial number *B*. If it detects *B*, the reader would refrain from scanning the privacy zone. This privacy zone respect would be directly auditable using a device that presents the reader with simulated tag sets and listens to the resulting reader signals. Soft blocking could support a flexible range of policies. For example, it could support an opt-in consumer privacy approach in which, by default, readers couldn't read private tags unless a special "deblocker" tag was present.

2.2.5 Policy Solutions

Even at this early stage, RFID privacy has attracted the keen attention of policymakers and legislators. The U.S. Federal Trade Commission has issued a report that addresses the impact of RFID on consumers, with an emphasis on privacy, but has not yet expressed an intention to issue regulations. EPCglobal Inc. has published guidelines for its members on EPC privacy for consumer products. These guidelines emphasize consumer education about the presence and functioning of EPC tags, and the provision of means of disablement or removal. Good public policies for RFID are likely to prove hard to craft,

because RFID tags, having essentially no form of access control, offer no obvious points of liability for information leakage. A healthcare provider, for instance, can issue a privacy policy describing the ways in which it grants or denies access to its customer databases; when a database is compromised, the target of liability is (more or less) clear. In contrast, a retailer cannot offer any guarantees about the tracking of RFID tags on items that leave its premises. RFID privacy is only fully meaningful if *all* entities with RFID readers subscribe to it – or if consumers do not carry live RFID tags. (It is of note that the EPCglobal guidelines do not really treat the problem of privacy in live RFID tags.) Given the inevitable deficiencies of technology or policy acting in isolation, the cooperation of technologists and legislators seems essential to good RFID privacy enforcement. Ontario's information and Privacy Commissioner has released privacy *Guidelines* for the growing field of radio frequency identification (RFID). EPCglobal Canada, an industry association that sets standards for electronic product codes, has been collaborating with the IPC in the development of this *Guideline.* The *Guidelines* address key privacy issues regarding the use of RFID technology at an item-level in the retail sector.

2.3 Cryptographic Solutions

Cryptography provides some measure of privacy for Basic tags. Storing encrypted serial numbers on tags initially seems like a viable approach to privacy protection. It introduces two problems, though. First, there's the problem of key management. How will the corresponding *decryption* key be distributed and managed? Second, simple encryption doesn't solve the tracking problem. An encrypted serial number is itself a kind of meta-serial number— namely, a static identifier that can be used to track an RFID tag's possessor.

Juels and Pappu [25] consider the special problem of consumer privacy-protection for RFID enabled banknotes. Their scheme employs a public-key cryptosystem with a single key pair: A public key PK, and a private key SK held by an appropriate law enforcement agency. An RFID tag in this system carries a unique identifier S, the banknote serial number. S is encrypted under PK as a cipher text C; the RFID tag emits C. Only the law enforcement agency, as possessor of the private key SK, can decrypt C

and thus learn the serial number S. To address the threat of tracking, Juel et al., propose that the cipher text C be periodically *re-encrypted.* They envisage a system in which shops and banks possess re-encrypting readers programmed with PK. In order to prevent wanton re-encryption by, e.g., malicious passersby, they propose that banknotes carry optical write-access keys; to re-encrypt a cipher text, a reader must scan this key. (RFID-enabled passports may employ a similar mechanism.) From several perspectives, like the need for re-encrypting readers, the JP system is very cumbersome. But it helpfully introduces the principle that cryptography can enhance RFIDtag privacy even when tags themselves cannot perform cryptographic operations.

Nowadays encryption of tag data is presumed to be performed by some external device, e.g., a point-of-sale device. More sophisticated approaches to privacy protection are possible if tags themselves can perform standard cryptographic operations like encryption. Could the tag itself perform dynamic, onboard cryptographic operations? Even if the tag itself can perform encryption, due to the exigencies of cost, basic RFID tags do not contain a sufficient amount of circuitry to do so, but the situation could change with the development of more compact ciphers.

RFID tags with richer security capabilities can perform symmetric-*key* (cryptographic one-way) functions. In a symmetric key function, a cryptographic hash function h has the special property that for a random bitstring M of sufficient length, it is infeasible to compute M from knowledge of the hashed value h (M) alone. Hashing involves no secret key (and is therefore only loosely called a symmetric-key function). In contrast, s*ymmetric-key encryption*, sometimes called *secret key encryption*, relies upon a secret key k. With this key, a message or *plaintext* M can be encrypted as a *cipher text* C. Only with knowledge of k is it feasible to decrypt C and recover M. In principle, symmetric-key cryptography can go far toward eliminating the problem of tag cloning with a simple challenge-response protocol. However, resource constraints in commercial RFID tags sometimes lead to the deployment of weak cryptographic primitives, and thus vulnerable authentication protocols.

The problems of key management and implementation of primitives are extremely important ones in this area. Just as important as the effective *use* of symmetric-key cryptographic primitives for privacy or authentication are the efficient design and

implementation of these primitives. A lightweight hardware implementation of a symmetric-key cipher, namely a 128-bit version of the AES (Advanced Encryption Standard) is proposed by Feldhofer et.al.,

This work is focused on implementation of an advanced cryptographic algorithm in more compact silicon. Realizing the AES (Advanced Encryption Standard) 128 algorithms in FPGA would enable prototyping of low cost implementation of Tags that can perform Cryptographic operations.

Chapter 3

AES Algorithm and FPGA Implementation

3.1 Why AES?

There are several cryptographic algorithms available. For example RC4 by Ron Rivest of RSA laboratories, Blow fish by Bruce Schnier and DES/ 3 DES which is a predecessor of AES (Advanced Encryption Standard). RC4 is a cipher with a key size of up to 2048 bits (256 bytes), which on the brief examination given it over the past year or so seems to be a relatively fast and strong cipher. It creates a stream of random bytes and 'XORing' those bytes with the text. It is useful in situations in which a new key can be chosen for each message and BLOWFISH is a symmetric block cipher just like DES. It takes a variable-length key, from 32 to 448 bits, making it ideal for both domestic and exportable use. Bruce Schneier designed Blowfish in 1993 as a fast, free alternative to the then existing encryption algorithms. Since then Blowfish has been analyzed considerably, and is gaining acceptance as a strong encryption algorithm. The Data Encryption Standard (DES) on the other hand was developed and endorsed by the U.S. government in 1977 as an official standard and forms the basis not only for the Automatic Teller Machines (ATM) PIN authentication but a variant is also utilized in UNIX password encryption. DES is a block cipher with 64-bit block size that uses 56-bit keys. Due to recent advances in computer technology, some experts no longer consider DES secure against all attacks; since then Triple-DES (3DES) has emerged as a stronger method. Using standard DES encryption, Triple-DES encrypts data three times and uses a different key for at least one of the three passes giving it a cumulative key size of 112-168 bits.

But the advancement in computer technology has assisted in the development of techniques to crack DES keys in meaningful time periods. In 1999 a distributed computing project managed to crack a DES key in 22 hours and 15 minutes. Both DES and 3 DES encryption is no longer considered strong enough for high security applications. AES keys can be 128, 192 or 256 bits long. Most commonly the 128 bit key length is used - giving a total of 3.4 x 10^38 possible keys. AES development has been a co-operative venture between the U.S. government and the private industry sector resulting in a encryption system that is royalty free. The effort to develop AES was started by NIST (National Institute for Standards and Technology), a US government organization. The intention was to develop a viable successor to the well-known Data Encryption Standard (DES) which has now been in use worldwide for a good 20 years. Five major candidates were submitted for the final round of evaluation: MARS, RC6, Rijndael, Serpent and Twofish, submitted by (respectively) IBM, Burt Kaliski (RSA labs), Johan Daemen (Banksys), Ross Anderson (University of Cambrigde) and Bruce Schneier (Counterpane Systems). Recently, NIST have announced that they have chosen Rijndael as the AES algorithm. This project is based on Rijndael AES.

3.2 AES Algorithm

The Advanced Encryption Standard (AES) is a computer security standard with the cryptography scheme is a symmetric block cipher that encrypts and decrypts 128-bit blocks of data. Lengths of 128, 192, and 256 bits are standard key lengths used by AES.

The algorithm consists of four stages that make up a round which is iterated 10 times for a 128-bit length key, 12 times for a 192-bit key, and 14 times for a 256-bit key. The first stage "SubBytes" transformation is a non-linear byte substitution for each byte of the block. The second stage "ShiftRows" transformation cyclically shifts (permutes) the bytes within the block. The third stage "MixColumns" transformation groups 4-bytes together forming 4-term polynomials and multiplies the polynomials with a fixed polynomial mod (x^4+1). The fourth stage "AddRoundKey" transformation adds the round key with the block of data. In most ciphers, the iterated transform (or round) usually has a Feistel Structure. Typically in this structure, some of the bits of the intermediate state are transposed unchanged to another position (permutation). AES

Algorithm does not have a Feistel structure but is composed of three distinct invertible transforms based on the Wide Trial Strategy design method. The Wide Trial Strategy design method provides resistance against linear and differential cryptanalysis.

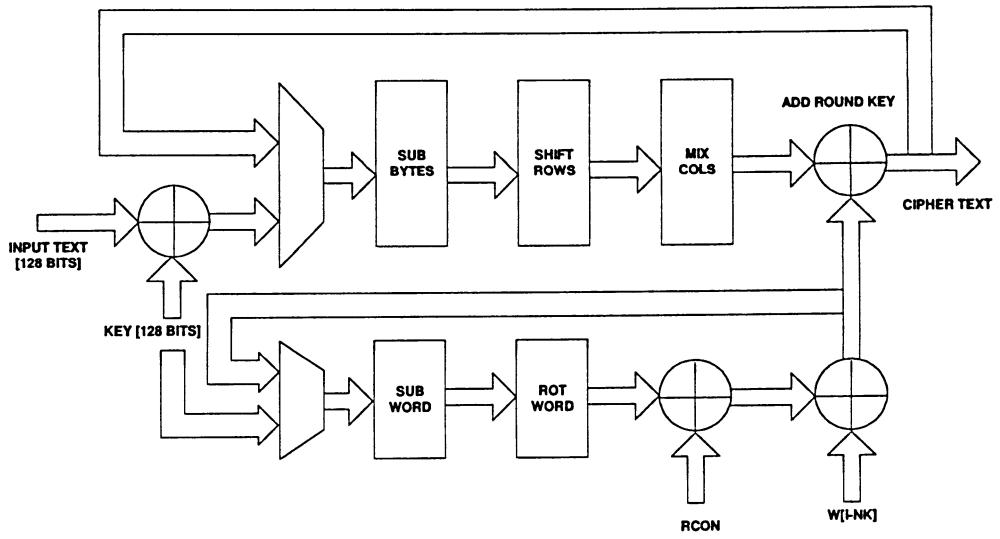


Figure 3.1: Architectural block diagram

3.2.1 SubBytes Transformation

The SubBytes transformation is a non-linear byte substitution that operates independently on each byte of the State using a substitution table (S-box). This S-box (Fig.3.2), which is invertible, is constructed by composing two transformations:

1. Take the multiplicative inverse in the finite algebric field (2^8); the element {00} is mapped to itself.
2. Apply the following transformation:

$$b_i' = b_i \oplus b_{(i+4) \bmod 8} \oplus b_{(i+5) \bmod 8} \oplus b_{(i+6) \bmod 8} \oplus b_{(i+7) \bmod 8} \oplus c_i$$

For $0 \le i < 8$, where bi is the i^{th} bit of the byte, and ci is the i^{th} bit of a byte c with the value {63} or {01100011}. Here and elsewhere, a prime on a variable indicates that the variable is to be updated with the value on the right.

The S-box used in the SubBytes transformation is presented in hexadecimal form in Fig. 3.2. For example, if $S_{1,1}$= {53}, then the substitution value would be determined by the intersection of the row with index '5' and the column with index '3' in Fig. 3.2. This would result in $S'_{1,1}$ having a value of {ed}.

		y														
		0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
x	0	63	7c	77	7b	f2	6b	6f	c5	30	01	67	2b	fe	d7	ab	76
	1	ca	82	c9	7d	fa	59	47	f0	ad	d4	a2	af	9c	a4	72	c0
	2	b7	fd	93	26	36	3f	f7	cc	34	a5	e5	f1	71	d8	31	15
	3	04	c7	23	c3	18	96	05	9a	07	12	80	e2	eb	27	b2	75
	4	09	83	2c	1a	1b	6e	5a	a0	52	3b	d6	b3	29	e3	2f	84
	5	53	d1	00	ed	20	fc	b1	5b	6a	cb	be	39	4a	4c	58	cf
	6	d0	ef	aa	fb	43	4d	33	85	45	f9	02	7f	50	3c	9f	a8
	7	51	a3	40	8f	92	9d	38	f5	bc	b6	da	21	10	ff	f3	d2
	8	cd	0c	13	ec	5f	97	44	17	c4	a7	7e	3d	64	5d	19	73
	9	60	81	4f	dc	22	2a	90	88	46	ee	b8	14	de	5e	0b	db
	a	e0	32	3a	0a	49	06	24	5c	c2	d3	ac	62	91	95	e4	79
	b	e7	c8	37	6d	8d	d5	4e	a9	6c	56	f4	ea	65	7a	ae	08
	c	ba	78	25	2e	1c	a6	b4	c6	e8	dd	74	1f	4b	bd	8b	8a
	d	70	3e	b5	66	48	03	f6	0e	61	35	57	b9	86	c1	1d	9e
	e	e1	f8	98	11	69	d9	8e	94	9b	1e	87	e9	ce	55	28	df
	f	8c	a1	89	0d	bf	e6	42	68	41	99	2d	0f	b0	54	bb	16

Figure 3.2: S-box in hexadecimal format (from [10])

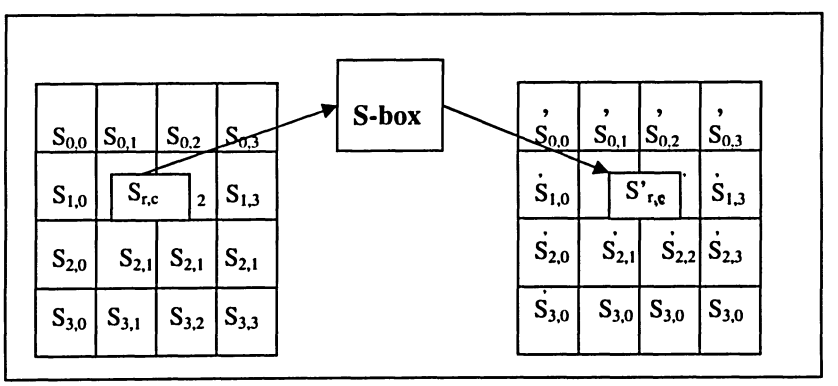


Figure 3.3: SubBytes applies the S-box to each byte of the State [10]

3.2.2 ShiftRows Transformation

In the ShiftRows transformation, the bytes in the last three rows of the State are cyclically shifted over different numbers of bytes (offsets). The first row, $r = 0$, is not shifted.

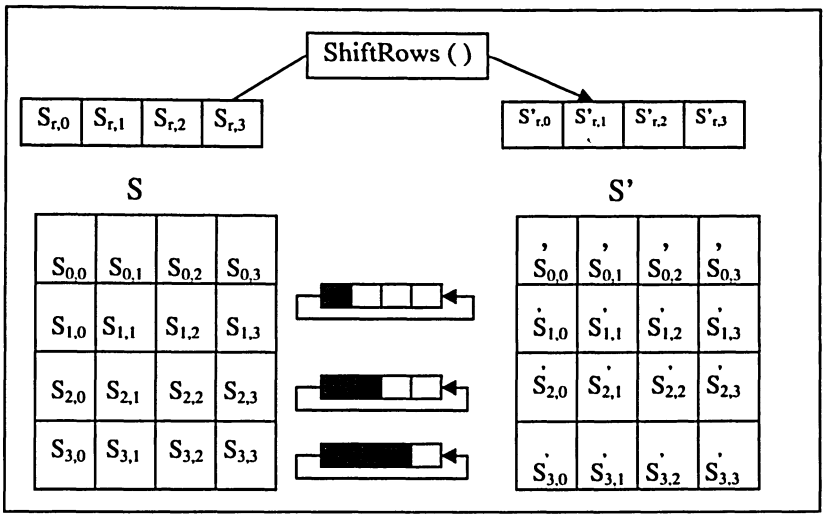


Figure 3.4: ShiftRows cyclically shifts the last three rows in the State [10]

3.2.3 MixColumns Transformation

The MixColumns transformation operates on the State column-by-column, treating each column as a four-term polynomial. The columns are considered as polynomials and multiplied modulo $x^4 + 1$ with a fixed polynomial $a(x)$, given by

$$a(x) = \{03\}x^3 + \{01\}x^2 + \{01\}x + \{02\}$$

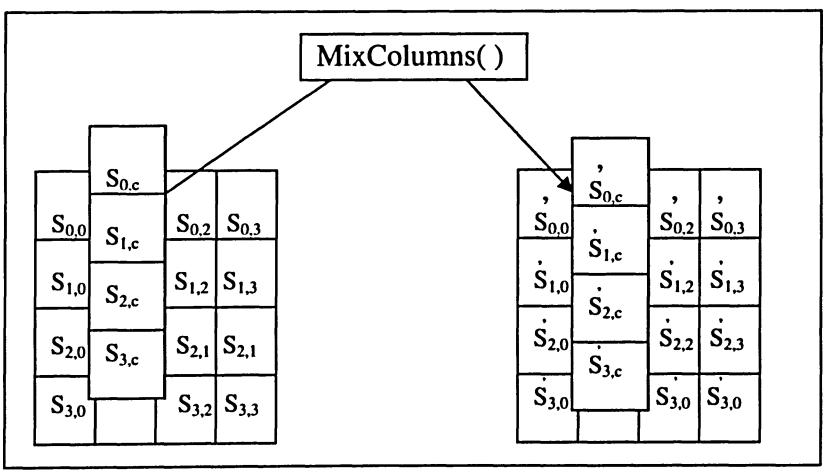


Figure 3.5: MixColumns operates on the State column-by-column [10]

3.2.4 AddRoundKey Transformation

In the AddRoundKey transformation, a Round Key is added to the State by a simple bitwise XOR operation. In the Cipher, the initial Round Key addition occurs when *round* = 0, prior to the first application of the round function.

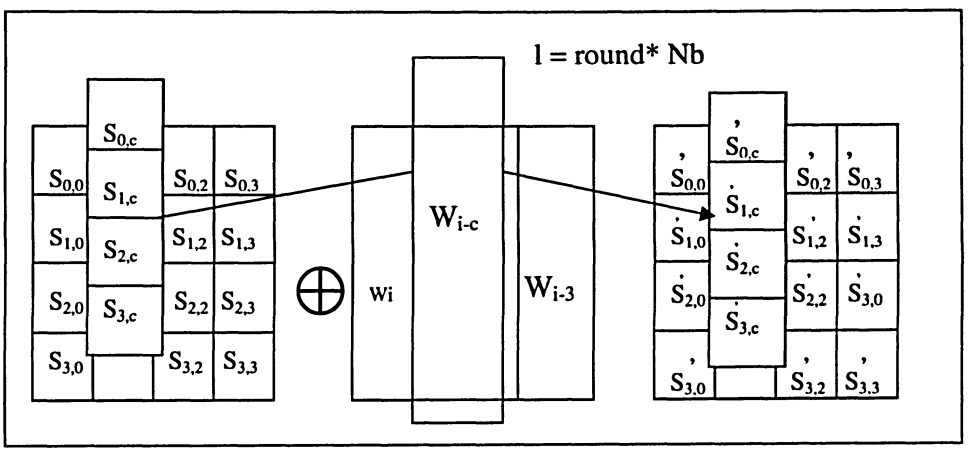


Figure 3.6: AddRoundKey XORs each column of the State with a word from the key schedule [10]

Once the hardware is developed to process one block, hardware duplication can be used to processes blocks in parallel allowing the AES cores to be customized for any application need. By duplicating hardware, the data rates can be increased linearly. Also by duplicating hardware with a constant data rate, the AES core clock-frequency can be reduced linearly allowing for a very low-power solution. All AES core configurations are below 50k gates and require around 512 bytes of ROM space. These low-cost cores are an ideal solution for RFID applications.

3.3 Strength of AES128 against known attacks

Despite the large amount of symmetry, care has been taken to eliminate symmetry in the behavior of the cipher. This is obtained by the round constants that are different for each round. The fact that the cipher and its inverse use different components practically eliminates the possibility for weak and semi-weak keys, as existing for DES (Double Encryption Standard). The non-linearity of the key expansion practically eliminates the possibility of equivalent keys.

DC(Differential Cryptanalysis) attacks are possible if there are predictable difference propagations over all but a few (typically 2 or 3) rounds that have a prop ratio (the relative amount of all input pairs that for the given input difference give rise to the output difference) significantly larger than 21-n if n is the block length. Difference propagation is composed of differential trails, where its prop ratio is the sum of the prop ratios of all differential trails that have the specified initial and final difference patterns. To be resistant against DC, it is therefore a necessary condition that there are no differential trails with a predicted prop ratio higher than 21-n. For AES 128, it's proved that there are no 4-round differential trails with a predicted prop ratio above 2–150 (and no 8-round trails with a predicted prop ratio above 2–300).

LC (Linear Cryptanalysis) attacks are possible if there are predictable input-output correlations over all but a few (typically 2 or 3) rounds significantly larger than 2n/2. An input-output correlation is composed of linear trails, where its correlation is the sum of the correlation coefficients of all linear trails that have the specified initial and final selection patterns. The correlation coefficients of the linear trails are signed and their sign depends on the value of the Round Keys. To be resistant against LC, it is a necessary condition that there are no linear trails with a correlation coefficient higher than 2n/2.For AES128, its proved that there are no 4-round linear trails with a correlation above 2 –75 (and no 8-round trails with a correlation above 2–150).

3.4 AES128 Algorithm Implementation

AES is implemented as defined in the FIPS-197 document in ECB mode. The decryption process follows virtually the same order as encryption except for another round of mix columns on the generated keys before giving them to the add round key step.

The encryption/decryption sequence:

Input data and key is fed in two blocks of 64 bits in consecutive clock cycles with the load signal. 64 bits of input and key are read in the posedge after the load signal goes high and another block of 64 bits of input and key are read in the posedge after the load signal goes low. Hence the complete data and key is loaded only when the load signal makes a low-high-low transition (basically a pulse). The process starts once the start

signal is pulsed and the output is validated with 'done' signal 13 clock cycles after the 'start' signal goes low. 'Done' remains high until the next start cycle.

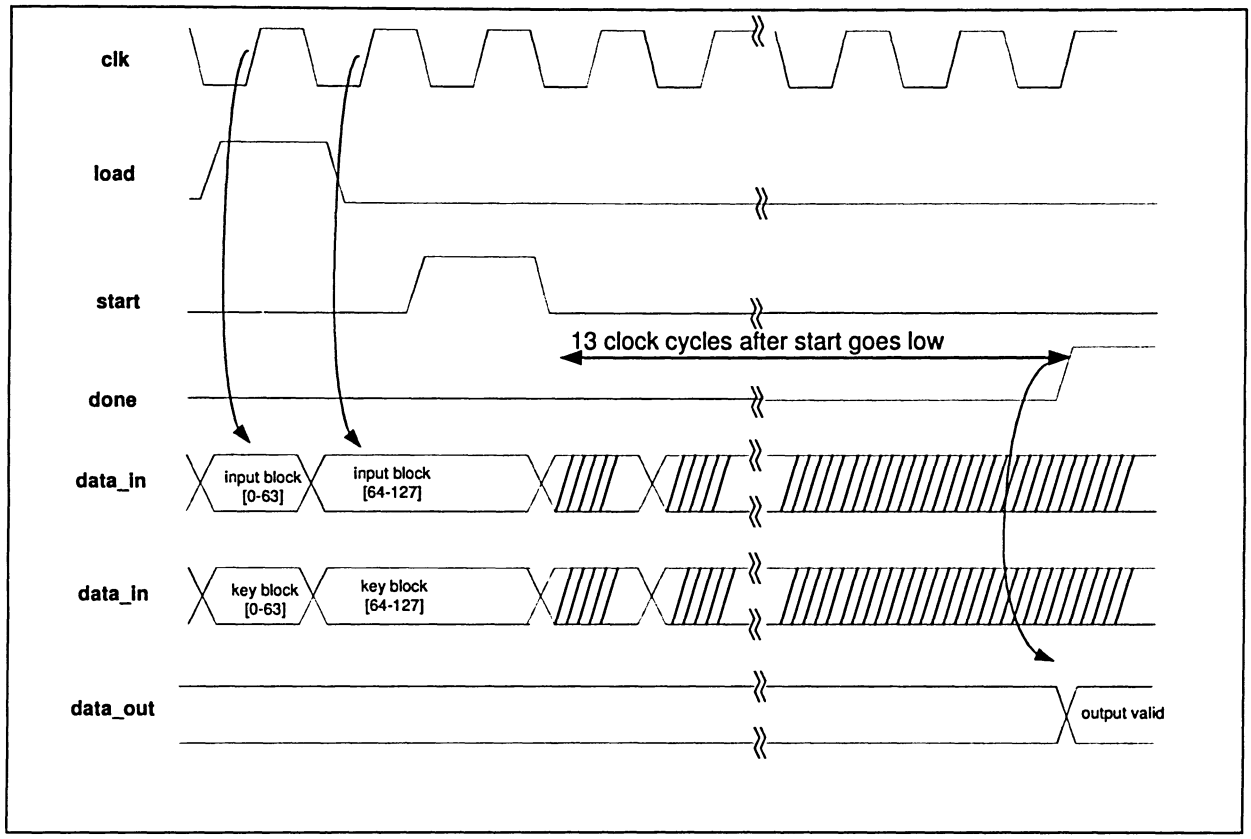


Figure 3.7: Process sequence for encryption/decryption

The architecture isn't pipelined and hence only able to perform multi encryption/decryption serially one after the other. The output is registered and available after 'done' signal goes high. Asserting the start signal during an encryption/decryption process will lead to erroneous output. The cipher key for the AES-128 algorithm is a sequence of 128 bits (can also be 192 or 256 bits for other algorithms). To transfer a 128 bit key or data block four write operations are necessary since the bus interface is 32 bit wide. After supplying a "key will be input" command to the control register, the key is input via four registers. After supplying a "data will be input" command to the control register, the input data is written via four registers. After the last input data register is

written, the encryption or decryption is started. The progress can be observed via the debug register. When the operation is completed, an interrupt is generated. The output data is then read out via four registers. It is not required to write a new key between each data input. There is no command needed for reading out the result.

The implementation requires around 89 clock cycles for a 128 bit data block in encryption direction and around 90 clock cycles for decryption direction. For decryption an initial key calculation is required. This takes around 10 additional clock cycles per every new key. Typically large amounts of data are decrypted (and also encrypted) with the same key. The key initialization for the decryption round does not influence the throughput.

3.5 Architecture

Basic architecture

The basic hardware architecture used to implement an encryption unit of a typical secret-key cipher is shown in Figure3.8. One round of the cipher is implemented as a combinational logic, and supplemented with a single register and a multiplexer. In the first clock cycle, input block of data is fed to the circuit through the multiplexer, and stored in the register. In each subsequent clock cycle, one round of the cipher is evaluated; the result is fed back to the circuit through the multiplexer, and stored in the register. The number of clock cycles necessary to encrypt a single block of data is equal to the number of cipher rounds, (referred as #rounds). We define the *speed* of the cipher implementation as the number of bits of data encrypted in a unit of time.

Speed calculated this way is often referred to as the circuit *throughput*. The speed of the basic architecture, $speed_{ba}$, is given by $speed_{ba} = 128 / \#rounds \cdot clock\ period$

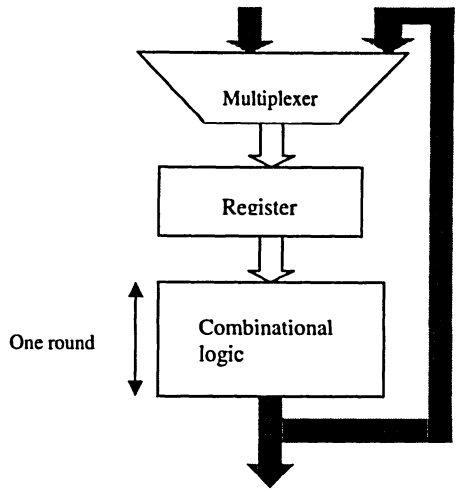


Figure 3.8: Basic Hardware Architecture

Alternate Architecture (Loop Unrolling)

Architecture with loop unrolling is shown in Figure 3.9. The only difference compared to the basic architecture is that the combinational part of the circuit implements k rounds of the cipher, instead of a single round. The maximum value of k is equal to the number of cipher rounds. The number of clock cycles necessary to encrypt a single block of data decreases by a factor of k. At the same time the minimum clock period increases by a factor slightly smaller than k, leading to an overall relatively small increase in the cipher speed, given by speedlu/speedba = (1 + t)/ (1+t/k), where t is the ratio of the sum of the multiplexer delay, the register delay and the register setup time to the delay of a single cipher round. This increase in speed is obtained at the cost of the circuit area. Because the combinational part of the circuit constitutes the majority of the circuit area, the total area of the encryption/decryption unit increases almost proportionally to the number of unrolled rounds, k. Additionally, the number of internal keys used in a single clock cycle increases by a factor of k, which in FPGA implementations typically implies the almost proportional growth in the number of CLBs used to store internal keys. In summary, loop unrolling enables increasing the circuit speed in both feedback and non-feedback operating modes. Nevertheless this increase is relatively small, and incurs a large area penalty.

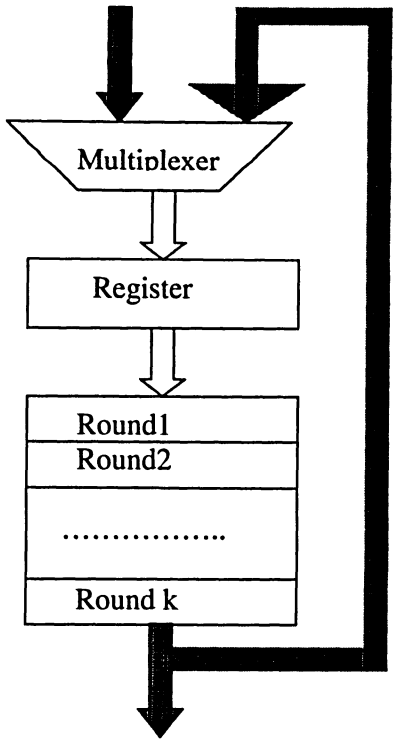


Figure 3.9: Loop Unrolling Architecture

3.6 FPGA Implementation

The latest generations of FPGAs featuring embedded processors offer compelling platforms for hardware acceleration of computationally intensive software algorithms. Taking advantage of these platforms, implementing AES algorithm in FPGA is a low-cost, low-risk platform for application prototyping as well as for use in high-performance end-products. This reconfigurable hardware technology, offers many advantages for future vendors and users of cryptographic equipment. It assures a short time to the market, high flexibility (including a capability for frequent modifications of hardware), low development costs, and low cost of the final product - the result of the algorithm agility - capability to use the same integrated circuit with time sharing for the execution of various secret-key and public key algorithms. Very few results regarding hardware implementations of the AES candidates have been published so far. So this project focuses on hardware implementation of AES algorithm in a FPGA that would help analyze the hardware implementation the security algorithms in RFID tags.

Basic organization of a block cipher implementation

The basic organization of the hardware implementation of a symmetric block cipher is shown in Figure 3.10. The organization includes the following units: a) *Encryption/decryption unit*, used to encipher and decipher input blocks of data b) *Key scheduling unit*, used to compute a set of internal cipher keys based on a single external key c) *Memory of internal keys*, used to store internal keys computed by the key scheduling unit, or loaded to the integrated circuit through the input interface d) *Input interface*, used to load blocks of input data and internal keys to the circuit, and to store input blocks awaiting encryption/decryption e) *Output interface*, used to temporarily store output from the encryption/decryption unit and send it to the external memory f) *Control unit*, used to generate control signals for all other units.

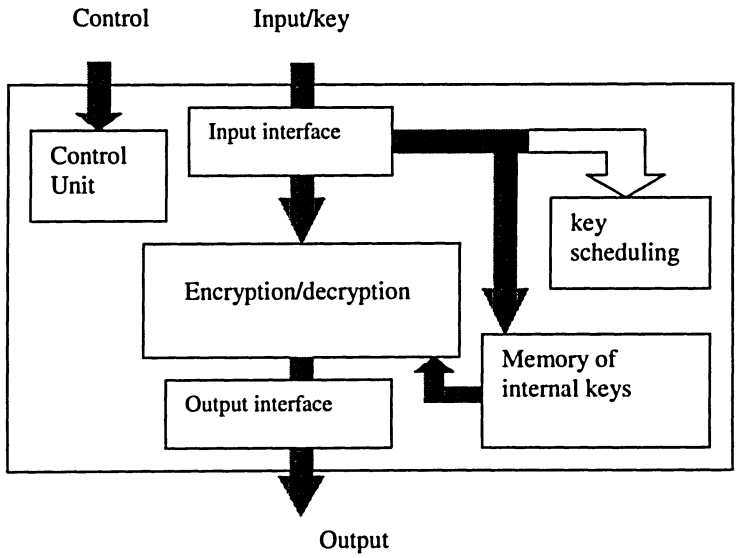


Figure 3.10: Block diagram of the hardware implementation of a symmetric-block cipher

Hardware is implemented for a fast prototype on an FPGA in VHDL.

3.7 Hardware Architecture

The AES 128 core consists of two blocks: 1) The AES Cipher block which performs encryption; 2) The AES Inverse Cipher block which performs decryption. Both blocks instantiate the same key expansion block.

3.7.1 AES Cipher Core

Below figure illustrates the overall architecture of the AES Cipher core.

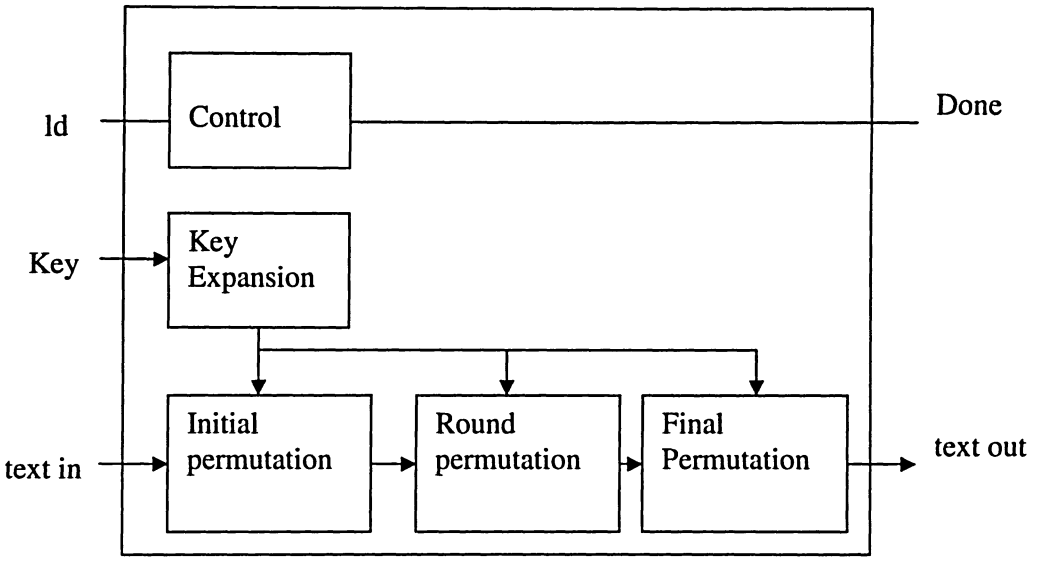


Figure 3.11: Cipher core Architecture overview

The AES cipher core consists of a key expansion module, an initial permutation module, a round permutation module and a final permutation module. The round permutation module will loop internally to perform 10 iteration (for 128 bit keys).

3.7.2 AES Inverse Cipher Core

Below figure illustrates the overall architecture of the AES Inverse Cipher core.

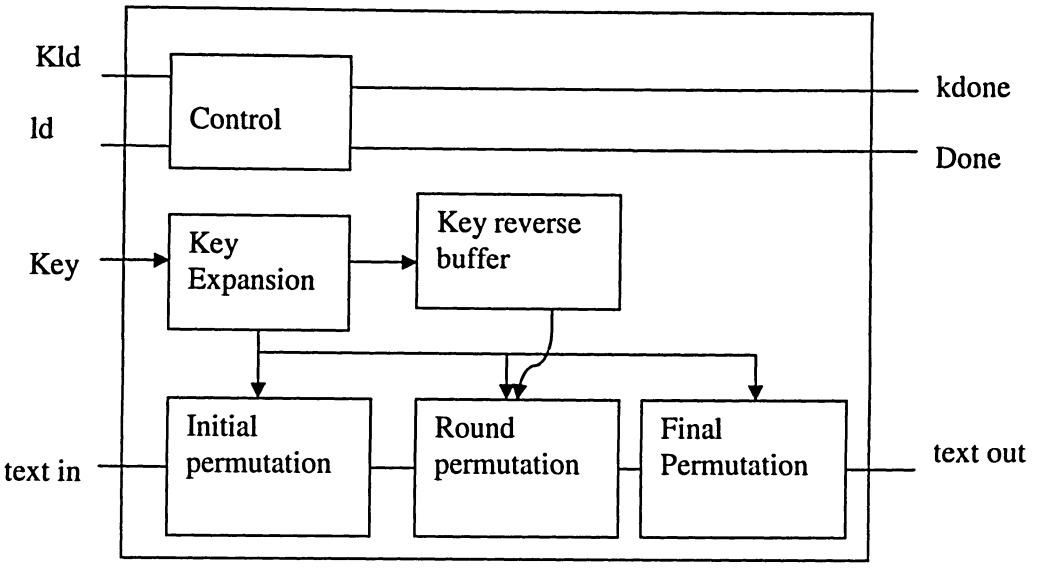


Figure 3.12: Inverse Cipher core Architecture overview

The AES inverse cipher core consists of a key expansion module, a key reversal buffer, an initial permutation module, a round permutation module and a final permutation module. The key reversal buffer first stores keys for all rounds and then presents them in reverse order to the inverse cipher rounds. The round permutation module will loop internally to perform 10 iteration (for 128 bit keys).

3.7.3 AES Cipher Core Operation

The forward cipher block can perform a complete encrypt sequence in 12 clock cycles (10 cycles for the 10 rounds, plus one cycle for initial key expansion, and one cycle for the output stage). The forward cipher block accepts a key and the plain text at the beginning of each encrypt sequence. The beginning is always indicated by asserting the 'ld' pin high. When the core completes the encryption sequence it will assert the 'done' signal for one clock cycle to indicate the completion. The user might chose to ignore the 'done' output and time the completion of the encryption sequence externally.

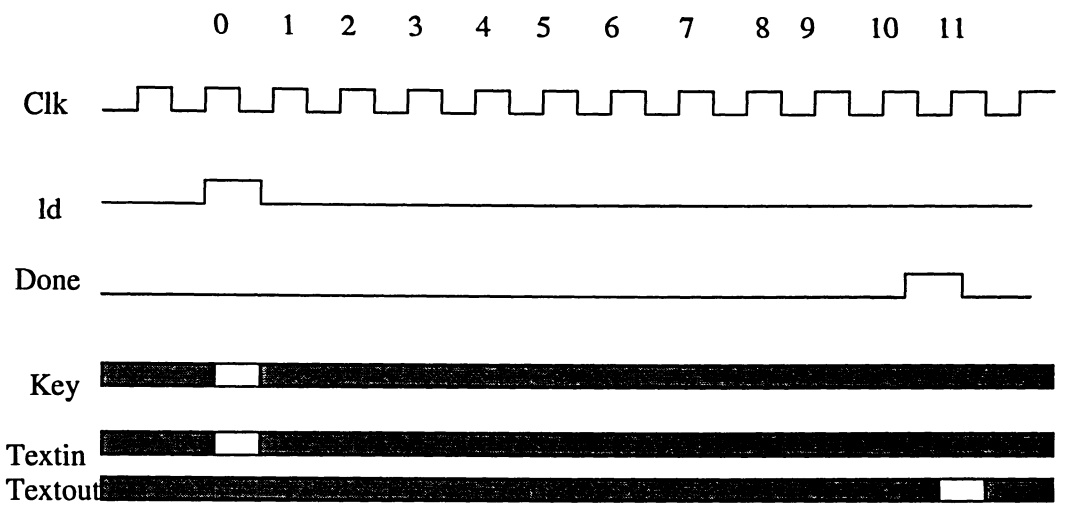


Figure 3.13: AES cipher core interface timing

3.7.4 AES Inverse Cipher Core Operation

The inverse cipher block can perform a complete decrypt sequence in 12 cycles (10 cycles for the 10 rounds, plus one cycle for initial key loading, and one cycle for the output stage). The inverse cipher, however, requires that the key is loaded before decryption can be performed. This is because it uses the last expanded key first and the first expanded key last. Once the key has been loaded, the expanded versions are generated and stored in an internal buffer. The expanded keys can be reused for subsequent decryption sequences for the same key. The key is loaded when the 'kld' signal is asserted high. Once key expansion sequence is completed, the 'kdone' signal will be asserted for one clock cycle. The beginning of the actual decrypt sequence is always indicated by asserting the 'ld' signal high. When the core completes the decryption sequence it will assert the 'done' signal for one clock cycle. The user might chose to ignore the 'done' output and time the completion of the decryption sequence externally. The key loading and decryption sequences can not happen in parallel. A key must always be loaded before the decryption sequence can be performed.

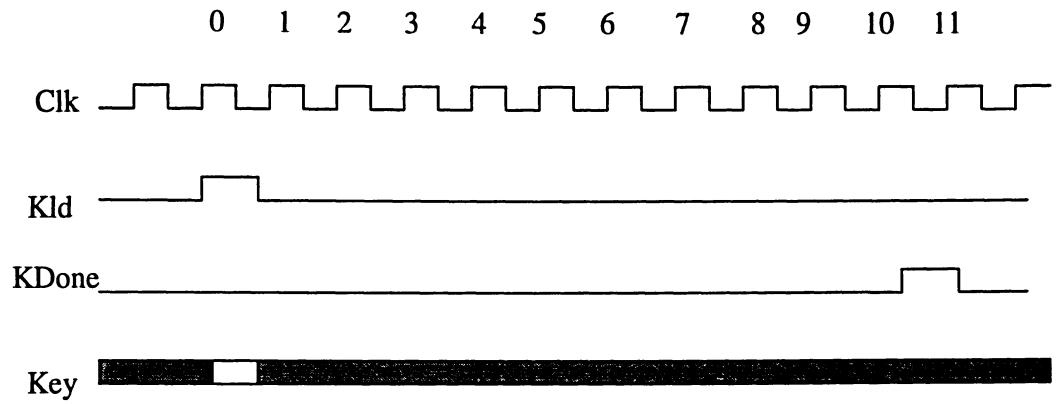


Figure 3.14: AES inverse cipher key load timing

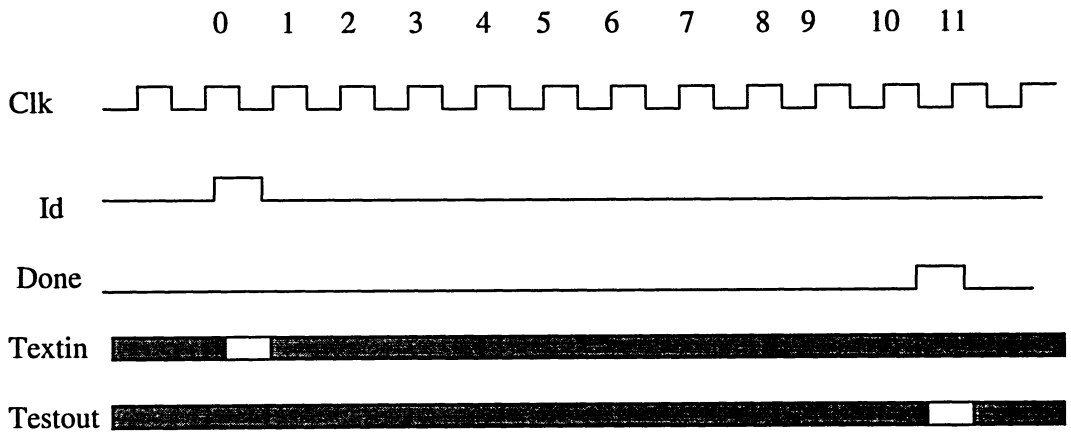


Figure 3.15: AES inverse cipher text block interface timing

Chapter 4

4.1 Implementation

The first step was to implement AES 128 algorithm in an Altera Stratix FPGA. A 'C' code for AES 128 core was taken from [47]. A VHDL code was derived from C core and compiled it using Quartus2 software. The compilation report and screen shots are shown in figure 4.1.

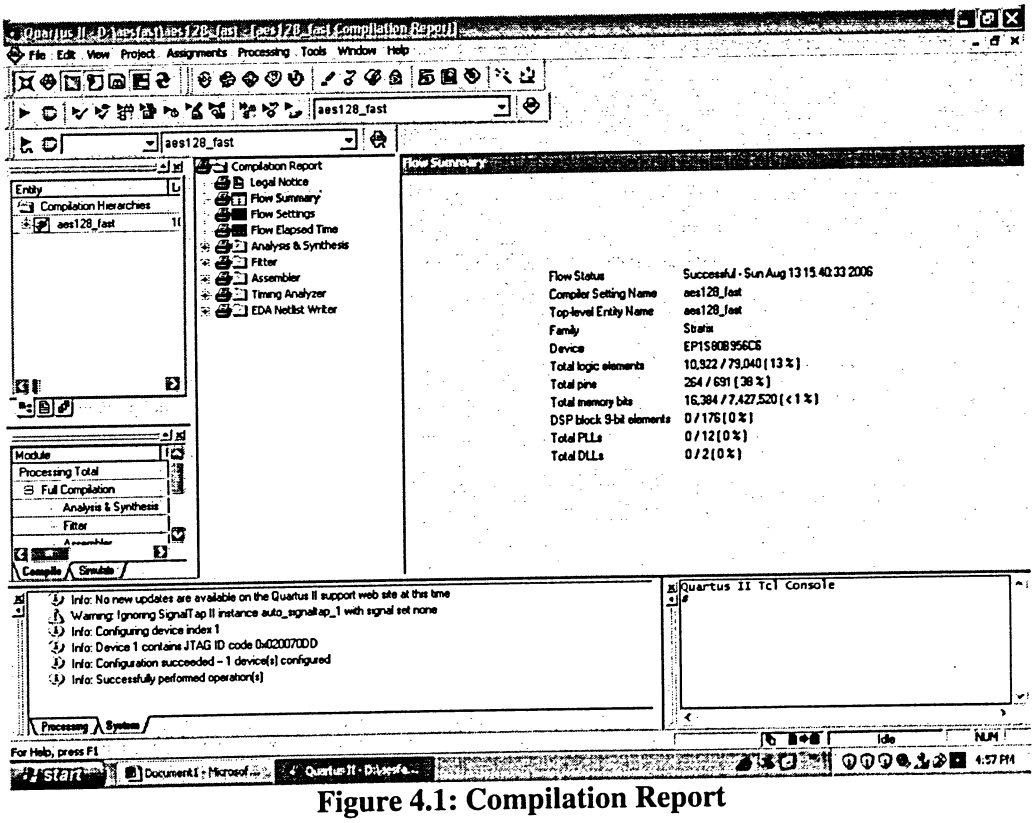
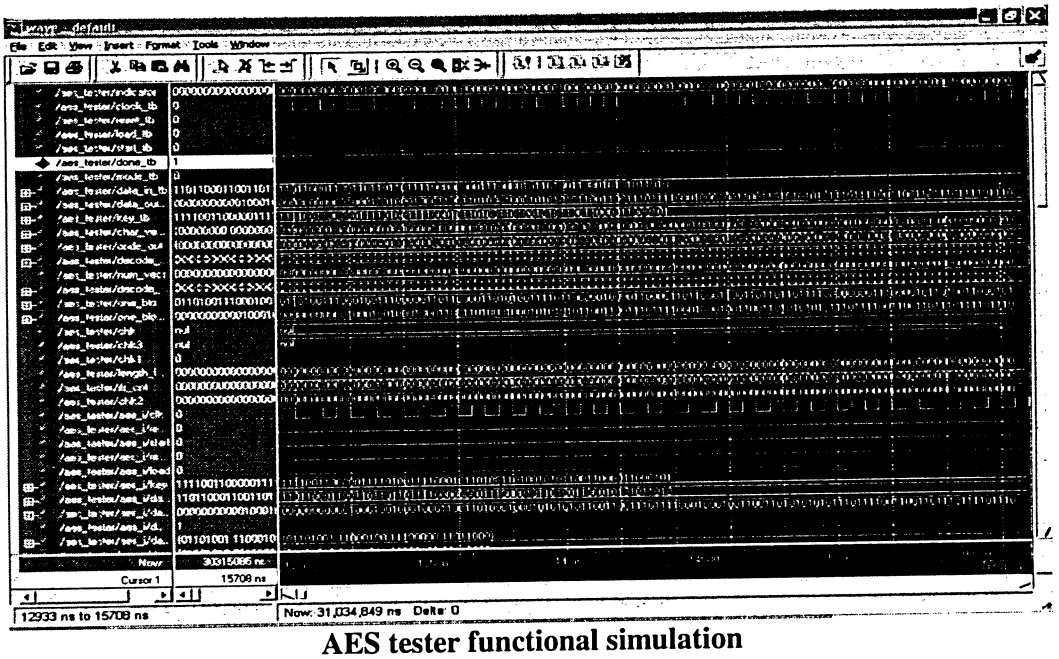


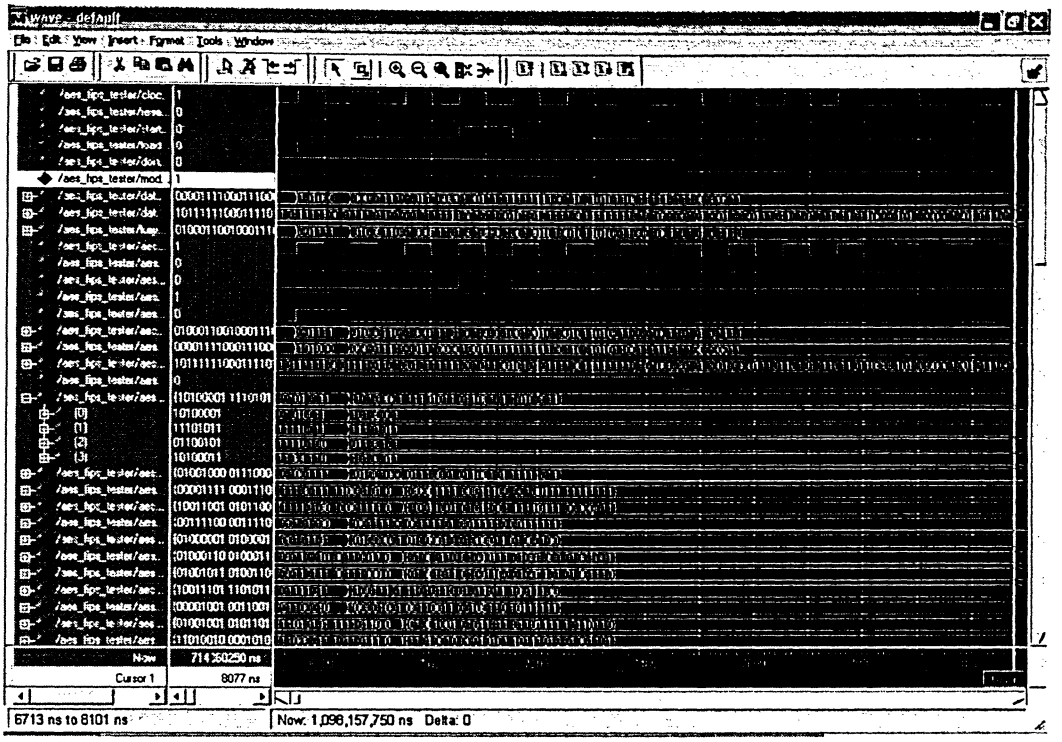
Figure 4.1: Compilation Report

Then the test bench simulation was done in Modelsim. As part of the testing, initial test benches were applied to verify the behavior of individual modules. The test

bench was built to test each of the modules using standard test vectors. The test benches made use of assertions, which allowed the correct functionality to be specified within the test bench. Using this approach, it was possible to run tests from the console, letting the test bench automatically verify outputs. The test bench simulation screen shots from Modelsim are shown below.



AES tester functional simulation



AES _fips tester functional simulation

Figure 4.2: Simulation of AES128

After successful execution of the test bench simulation, the AES 128 algorithm was programmed into Altera stratix FPGA.

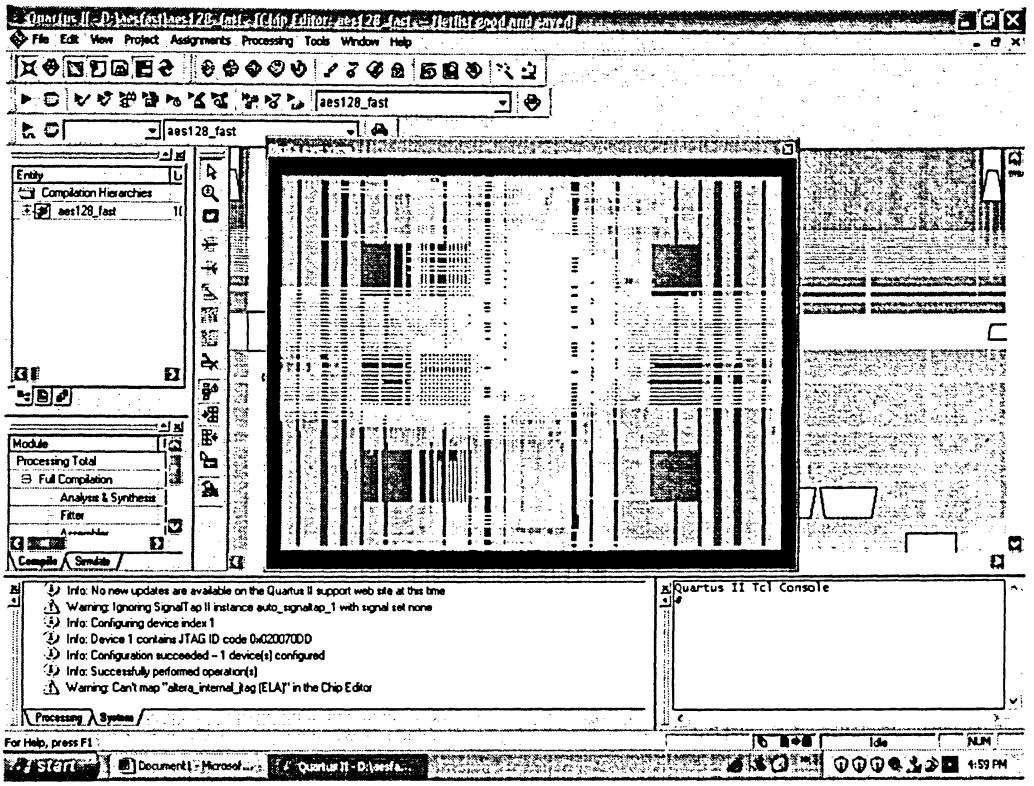


Figure 4.3: AES 128 Programmed

Once the AES 128 was programmed into the FPGA, the real challenge was to test it with real data from the RFID reader. The Texas Instruments S4100 Multi-Function Reader Module was used to read following RFID tags. Tag-it™ HF-I Standard Inlays, Tag-it™ ISO Vicinity Cards (13.56MHz) and Low Frequency Keyring Tag, 23mm Glass Transponders (134.2 kHz) were used by the Tag tracker Application to get the data from the tag. And these data fetched through the reader was stored in a text file called My_file.txt and fed into the FPGA using the RS232 utility.

The interface between the RFID reader and the AES128 entity in FPGA was emulated through the RS232 utility. The RS 232 utility was programmed and used to send data from RFID reader and receive data from the FPGA entity.

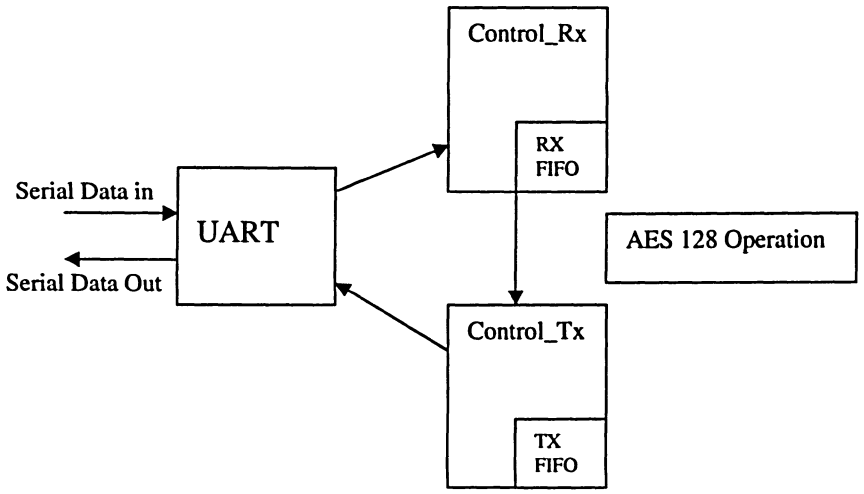
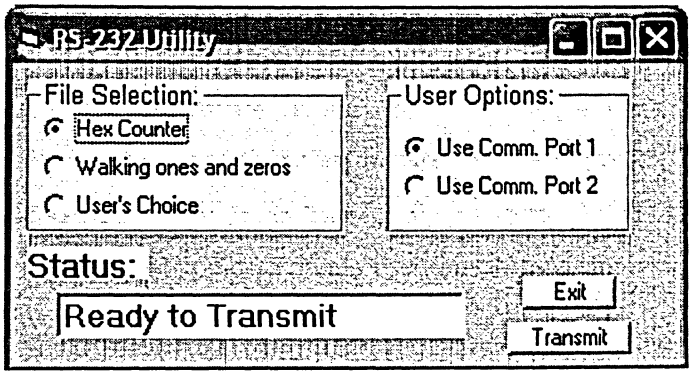


Figure 4.4: RS232 Utility Design

The RS232 utility design reads the data out of the RX FIFO buffer, do whatever processing is desired (in this case AES encryption & decryption), and then write the data back to the TX FIFO buffer. SW1 on the board controls the transfer of data from the RX FIFO buffer to the TX FIFO buffer. This switch also enables AES 128 encryption and decryption function. SW2 controls the transfer of data from the TX FIFO buffer to the PC.

Then, the Visual Basic Application sends the data to the development board. The universal asynchronous receiver/transmitter (UART) and Control_RX logic stores that data in the RX_FIFO buffer. Pressing SW1 initializes the AES processing function. When processing completes and all data resides in the TX_FIFO buffer, pressing SW2 sends the data back to the PC.

Ready to send



Data Sent

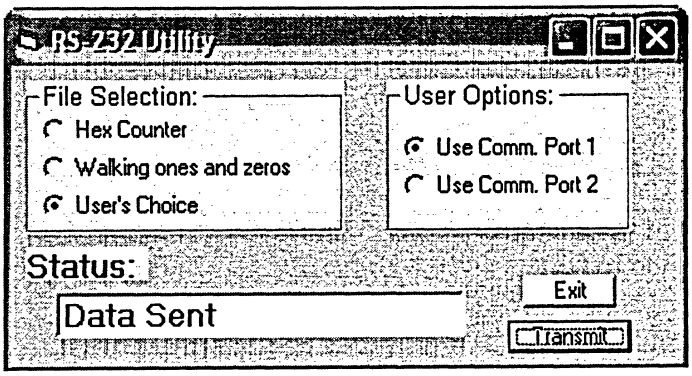


Figure 4.5: Data sent through RS232 utility

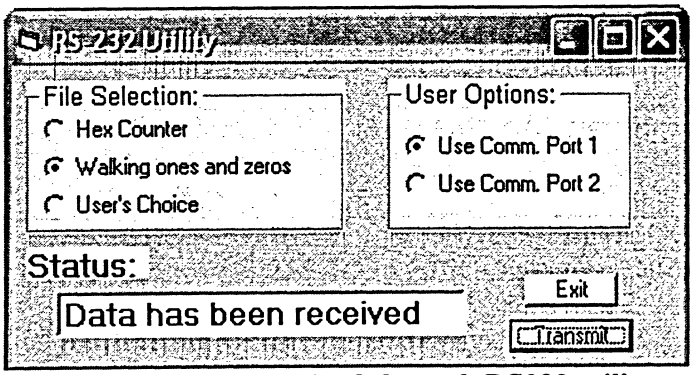


Figure 4.6: Data received through RS232 utility

4.2 Results

The successful implementation AES 128 algorithm in FPGA was confirmed by sending the text file My_File.txt using the RS232 VB application. The data sent is encrypted and decrypted and the result is obtained in the text file Data_back.txt. This is done by pressing switch SW1 on the board to initiate transfer of data from RX FIFO to the AES 128 operation. And switch SW2 initiates transfer of data from TX FIFO to PC in the Data back text file. The number of logic elements used and timing analysis screen shots for this operation is given in Appendix A.

The critical path includes one or several S-boxes, and several multiple-input XORs. The minimum clock period is the sum of the access time to memories used to implement S-boxes, and delays introduced by multiple-input XORs and other simple auxiliary operations. The effect of resource sharing between encryption and decryption on the critical path is very small for AES 128.

Table 4.1: Critical path in the implementation of AES 128 in Stratix FPGA

Cipher	Minimum Clock Period – Stratix [ns]	Number of Rounds	Components in Critical path (list of operations)
AES 128	16.2	10	S-box 8x8, XOR6, XOR5, XOR4, XOR2, 2 MUX2

Table 4.2: Cipher components contributing most to the circuit area.

Cipher	# of Logic Cells – Stratix	Area Critical Components
AES 128	2578	16 S-box 8x8 (32 kbit), 24 MUL GF(28), 256 XOR5 (affine and inverse affine transformation)

The total logic elements required for this implementation is 10922 in Stratix FPGA which can be further reduced by pipelining the algorithm. AES 128 is relatively easy to pipeline, but its critical path contains only 7 elementary operations. Additionally,

the most time-consuming of these operations, the 8x8 S-box read-out, is hard to divide into extra pipeline stages.

This shows implementation of security algorithm in a RFID Smart tag is possible. An implementation AES128 algorithm with the authentication protocol by Martin Feldoher in [9] would form a novel security layer for passive RFID Tags with restrictions in die size and power consumptions. The authentication protocol is a Simple Authentication and Security Layer *(SASL)* protocol as shown below in figure 4.7. A 128-bit random number *r* is generated by the reader and sent to the tag within a request frame. The tag encrypts this random number and sends it back to the reader within a response frame. The reader decrypts the received data and compares it with the sent data. If they are equal the reader can be sure of the authenticity of the tag.

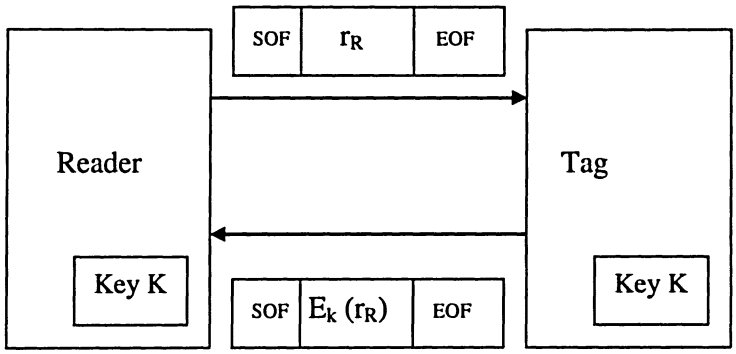


Figure 4.7: Proposed authentication protocol implementation by Martin Feldhofer

Chapter 5

5.1 Conclusion

This report started with an introduction to the RFID system and the motivation for this project. Then a survey on privacy and security of RFID system was given before discussing the AES cryptographic algorithm for RFID security. The main contribution is that secure symmetric authentication is feasible for current RFID technology without significant additional cost. Since RFID tags do not yet have cryptographic components, the implementation of AES128 in FPGA provides a good understanding of hardware implementation of cryptographic algorithms in RFID smart tags. This implementation shows that it is possible to integrate security aspects in RFID tags.

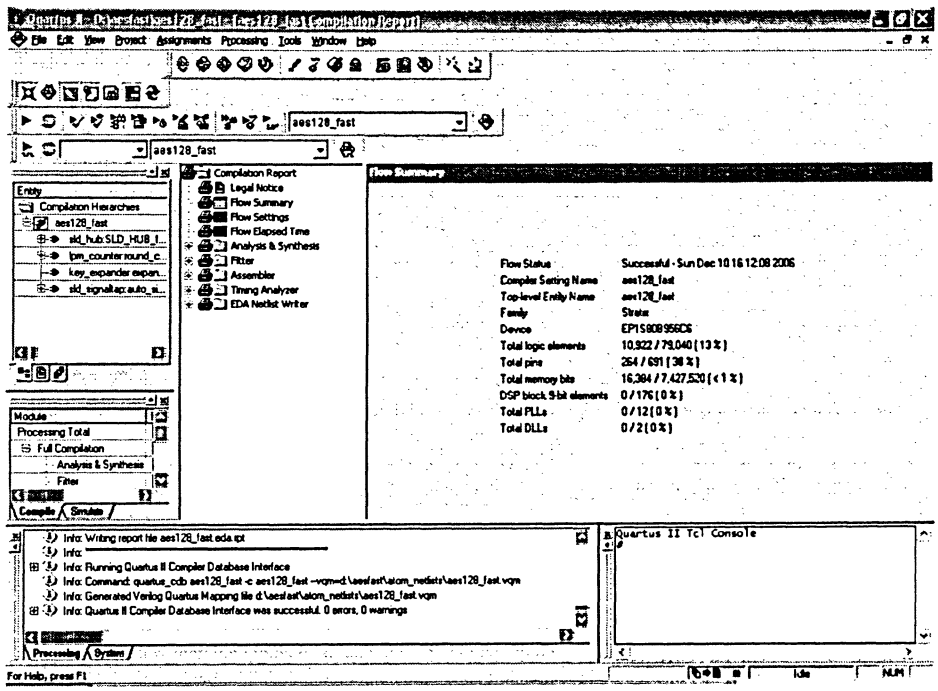
5.2 Future Work

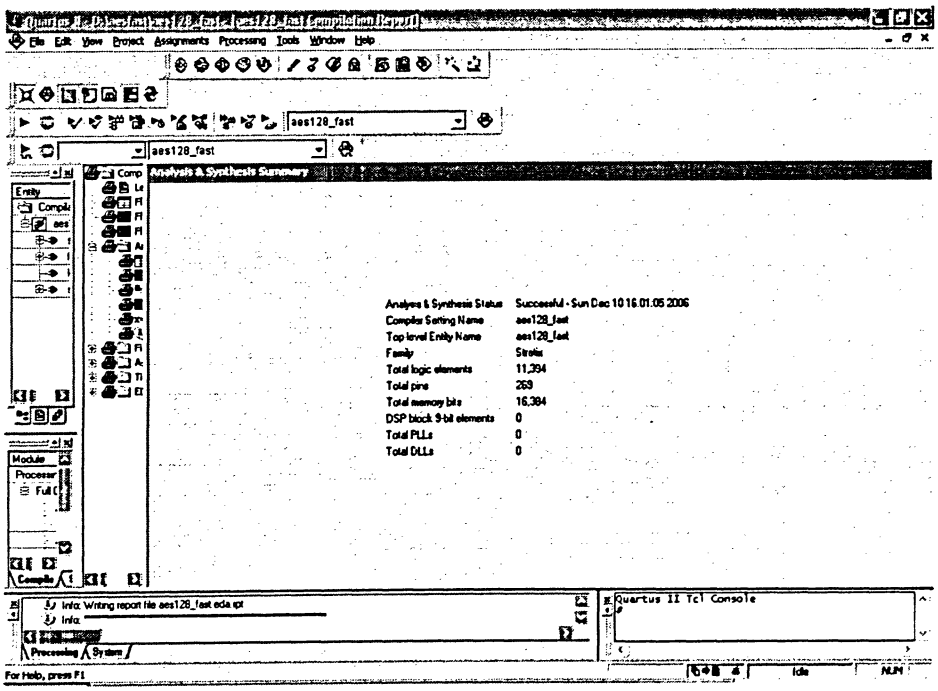
Future work may consist of power estimations using power simulation and building a proof-of-concept implementation using an FPGA. Also an integration of Hardware/Software implementation can be done using SoC systems where programmable logic is mixed with a CPU. Implementation with different hardware in terms of Timing Analysis and Power Consumption will provide in-depth knowledge of what kind of hardware to choose of RFID Tags with cryptographic algorithm.

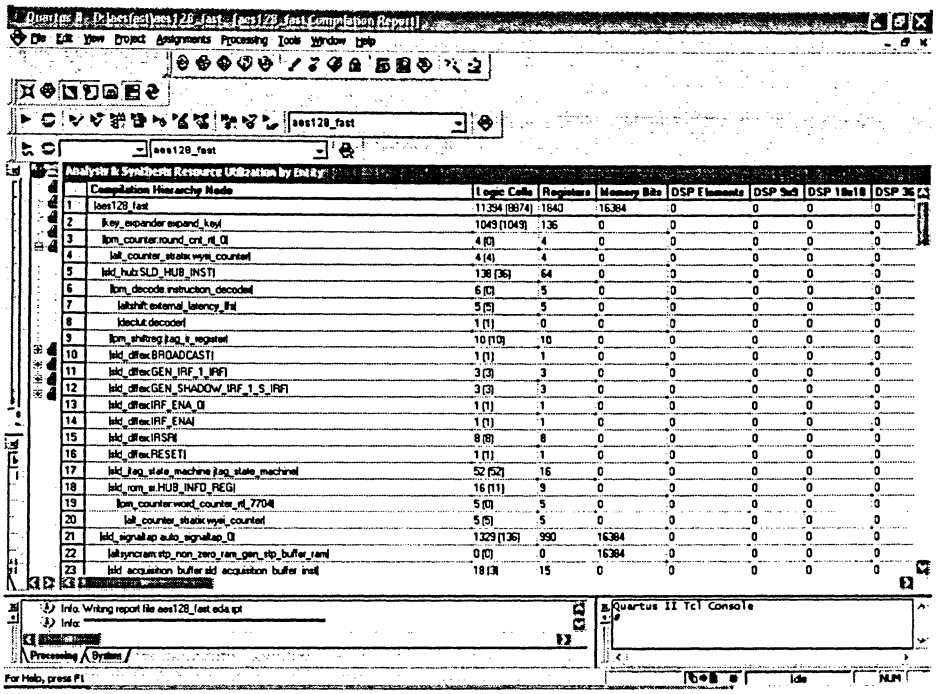
Apart from the improvements in FPGA implementation of AES algorithm, the future work can also look into following questions. Is it possible to construct a fully privacy-preserving, symmetric-key RFID identification scheme in which the reader performs computation? Alternatively, is it possible to prove that such a scheme is impossible without the use of public-key cryptography? Nearly all of the schemes discussed in this report presume a centralized model, namely that readers have continuous access to a centralized database. This feature provides resistance to replay and desynchronization attacks. What is the best way to engineer a system in which readers have only intermittent connectivity? Indeed, what is the best way to model such a system?

Appendix A

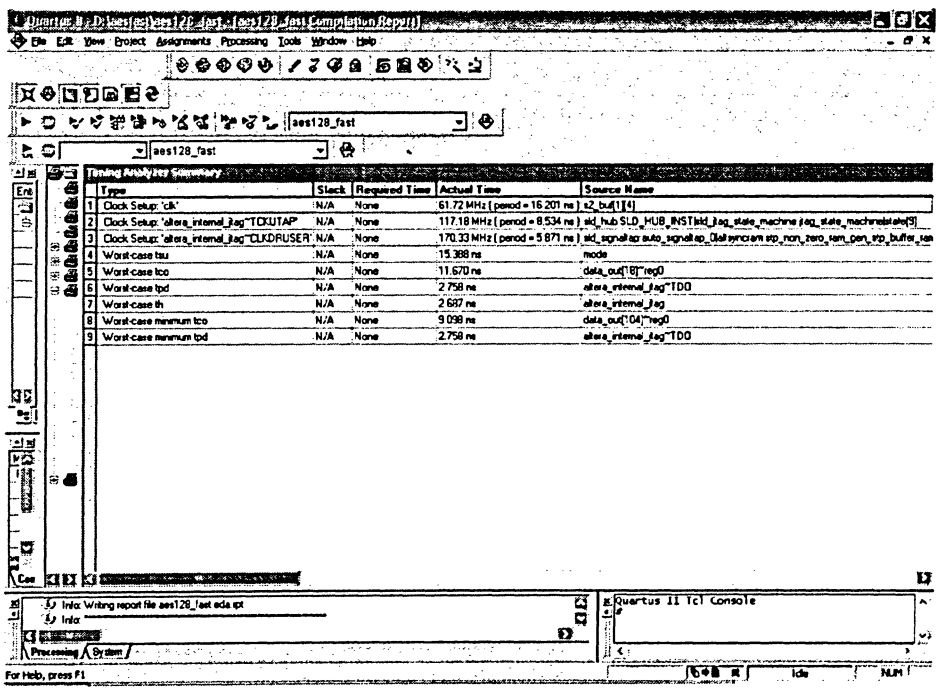
Compilation report and screen shots of Analysis and synthesis resource utilization.





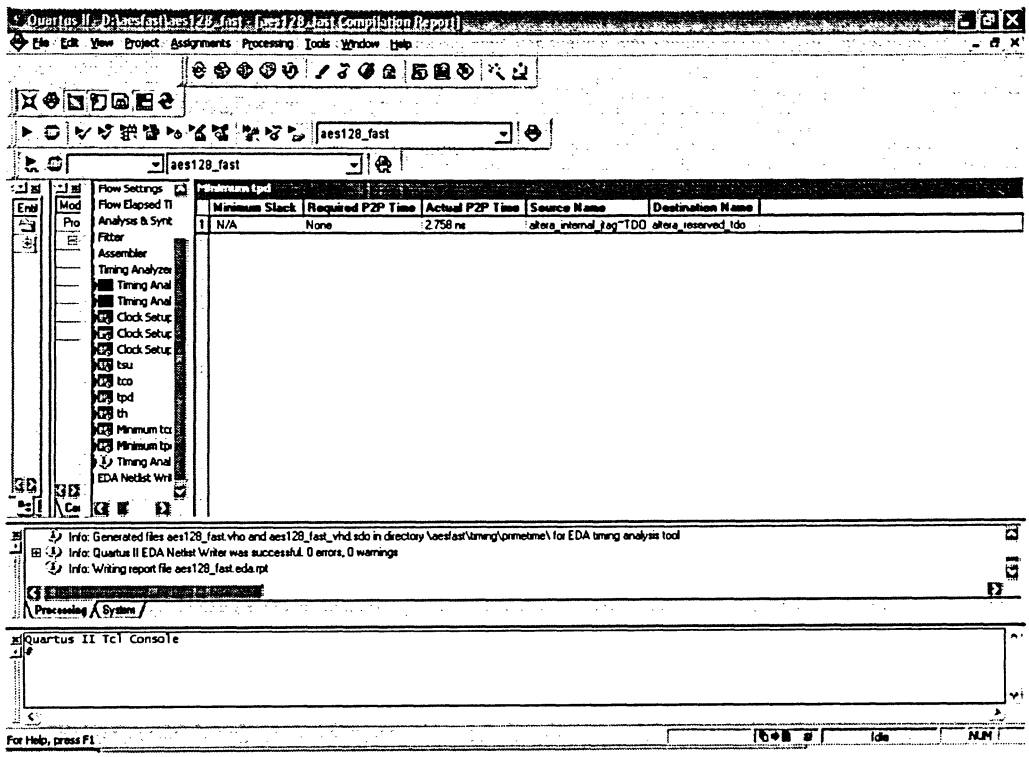


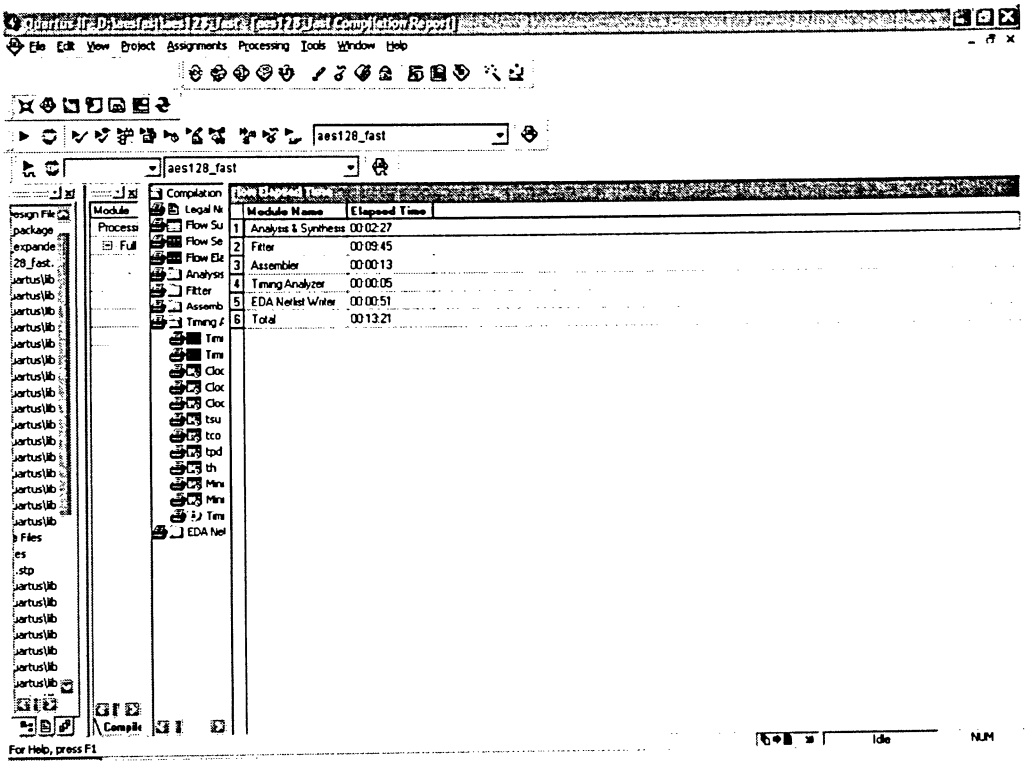
	Compilation Hierarchy Node	Logic Cells	Registers	Memory Bits	DSP Elements	DSP 9x9	DSP 18x18	DSP 36
1	aes128_fast	11394 (9874)	1840	16384	0	0	0	0
2	|key_expander:expand_key|	1049 (1049)	136	0	0	0	0	0
3	|lpm_counter:round_cnt_rtl_0|	4 (0)	4	0	0	0	0	0
4	|alt_counter_stratix:wys_counter|	4 (4)	4	0	0	0	0	0
5	|sld_hub:SLD_HUB_INST|	138 (36)	64	0	0	0	0	0
6	|lpm_decode:instruction_decoded|	6 (0)	5	0	0	0	0	0
7	|altshift:external_latency_ffs|	5 (5)	5	0	0	0	0	0
8	|declut:decoder|	1 (1)	0	0	0	0	0	0
9	|lpm_shiftreg:jtag_ir_register|	10 (10)	10	0	0	0	0	0
10	|sld_dffex:BROADCAST|	1 (1)	1	0	0	0	0	0
11	|sld_dffex:GEN_IRF_1_IRF|	3 (3)	3	0	0	0	0	0
12	|sld_dffex:GEN_SHADOW_IRF_1_S_IRF|	3 (3)	3	0	0	0	0	0
13	|sld_dffex:IRF_ENA_0|	1 (1)	1	0	0	0	0	0
14	|sld_dffex:IRF_ENA|	1 (1)	1	0	0	0	0	0
15	|sld_dffex:IRSR|	8 (8)	8	0	0	0	0	0
16	|sld_dffex:RESET|	1 (1)	1	0	0	0	0	0
17	|sld_jtag_state_machine:jtag_state_machine|	52 (52)	16	0	0	0	0	0
18	|sld_ram_s:HUB_INFO_REG|	16 (11)	9	0	0	0	0	0
19	|lpm_counter:word_counter_rtl_7704|	5 (0)	5	0	0	0	0	0
20	|alt_counter_stratix:wys_counter|	5 (5)	5	0	0	0	0	0
21	|sld_signaltap:auto_signaltap_0|	1329 (136)	990	16384	0	0	0	0
22	|altsyncram:stp_non_zero_ram_gen_stp_buffer_ram|	0 (0)	0	16384	0	0	0	0
23	|sld_acquisition_buffer:sld_acquisition_buffer_inst|	18 (3)	15	0	0	0	0	0

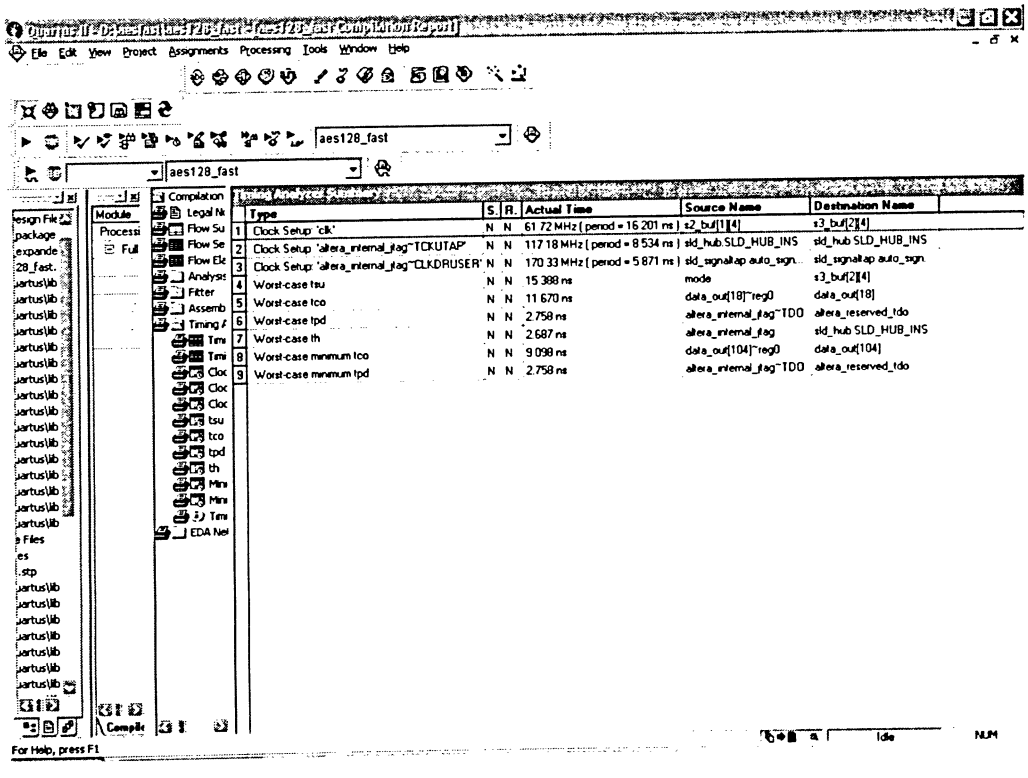


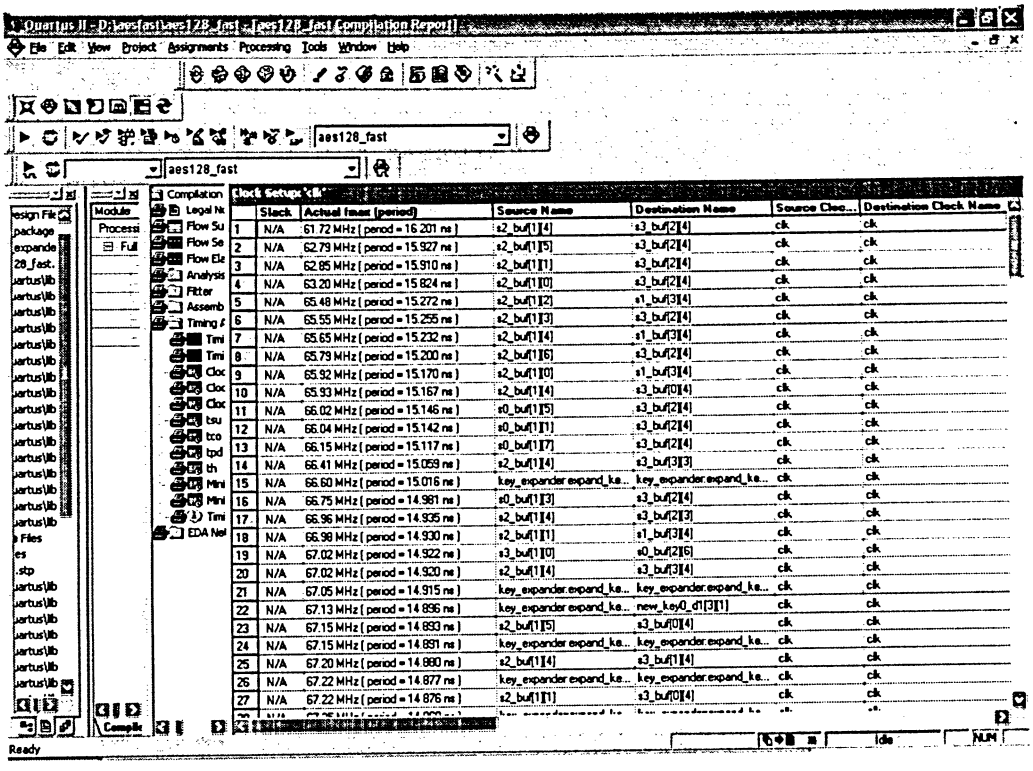
	Type	Slack	Required Time	Actual Time	Source Name
1	Clock Setup: 'clk'	N/A	None	61.72 MHz (period = 16.201 ns)	s2_buf[1][4]
2	Clock Setup: 'altera_internal_jtag~TCKUTAP'	N/A	None	117.18 MHz (period = 8.534 ns)	sld_hub:SLD_HUB_INST|sld_jtag_state_machine:jtag_state_machine|state[5]
3	Clock Setup: 'altera_internal_jtag~CLKDRUSER'	N/A	None	170.33 MHz (period = 5.871 ns)	sld_signaltap:auto_signaltap_0|altsyncram:stp_non_zero_ram_gen_stp_buffer_ram
4	Worst-case tsu	N/A	None	15.388 ns	mode
5	Worst-case tco	N/A	None	11.670 ns	data_out[18]~reg0
6	Worst-case tpd	N/A	None	2.758 ns	altera_internal_jtag~TDO
7	Worst-case th	N/A	None	2.687 ns	altera_internal_jtag
8	Worst-case minimum tco	N/A	None	9.098 ns	data_out[104]~reg0
9	Worst-case minimum tpd	N/A	None	2.758 ns	altera_internal_jtag~TDO

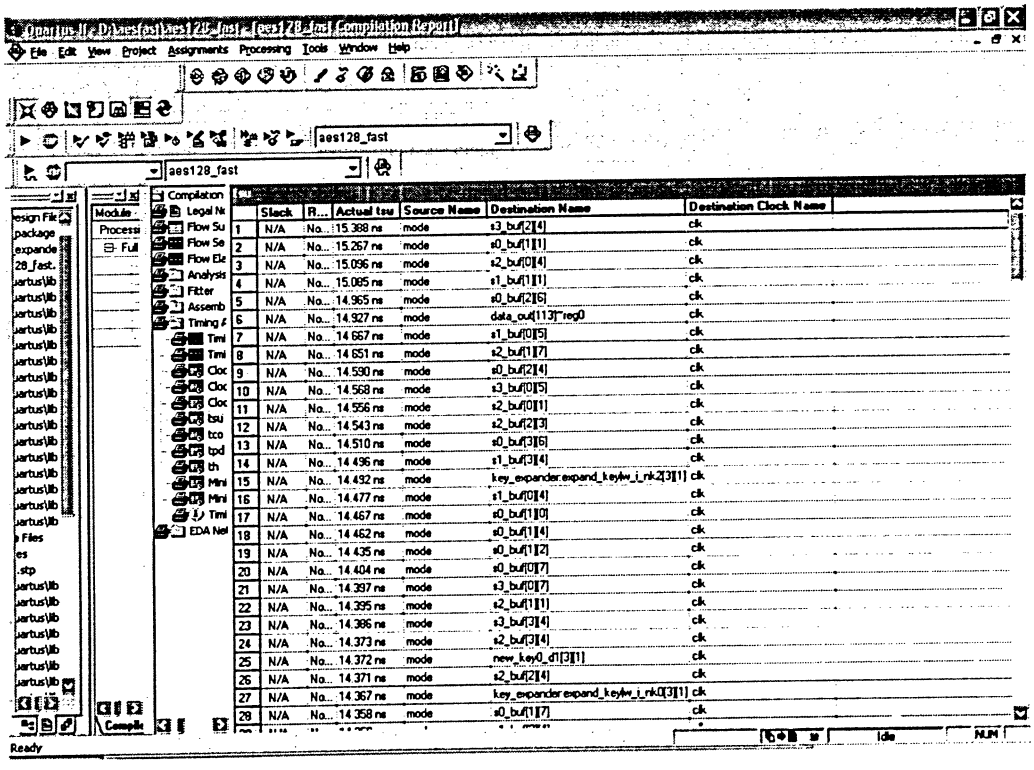
Timing Analysis Report Screen shots

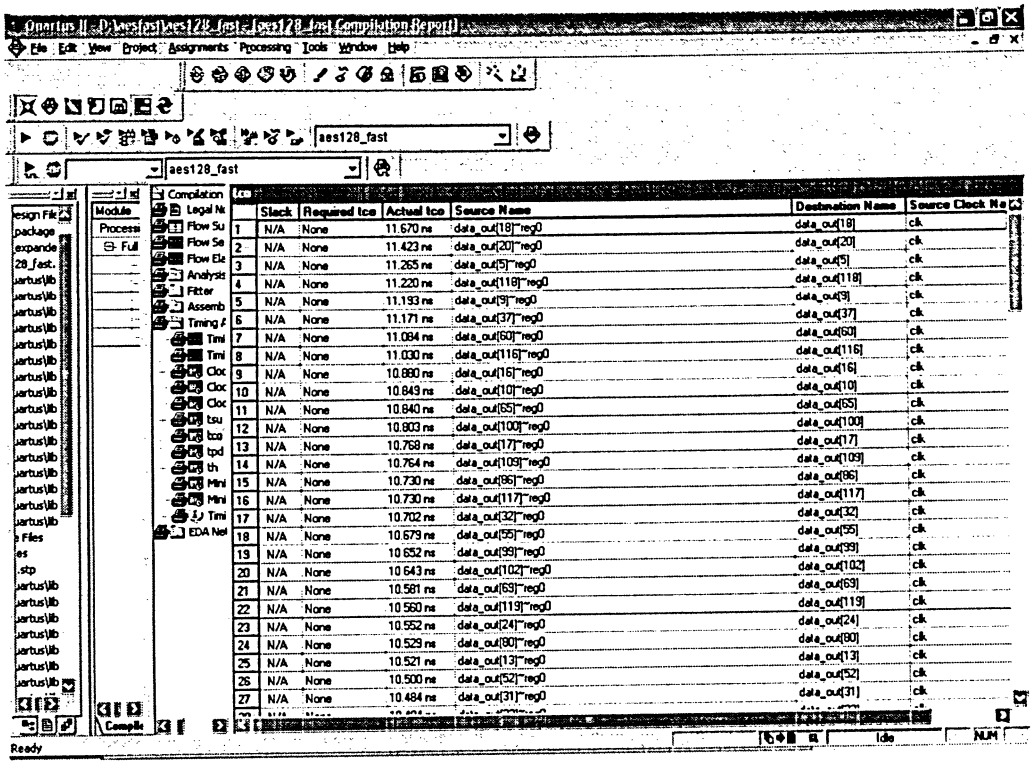


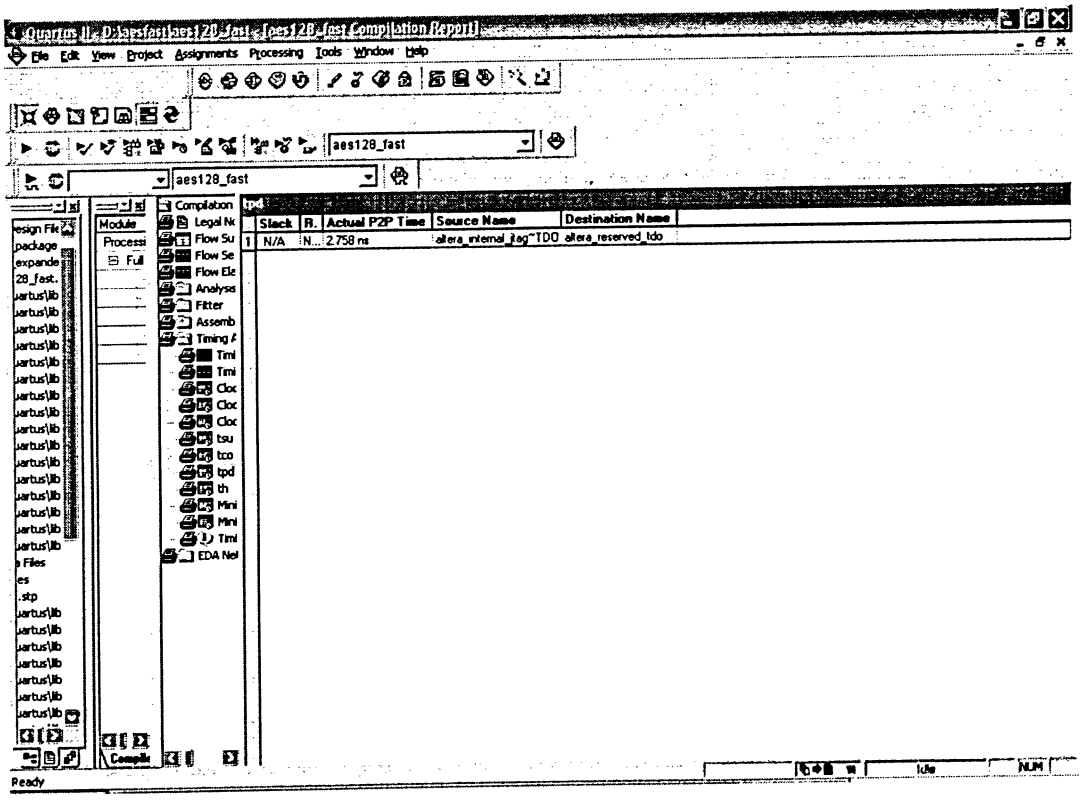

File Edit View Project Assignments Processing Tools Window Help
aes128_fast
aes128_fast
Module Name
Elapsed Time
1 Analysis & Synthesis 00:02:27
2 Fitter 00:09:45
3 Assembler 00:00:13
4 Timing Analyzer 00:00:05
5 EDA Netlist Writer 00:00:51
6 Total 00:13:21
For Help, press F1
Idle
NUM

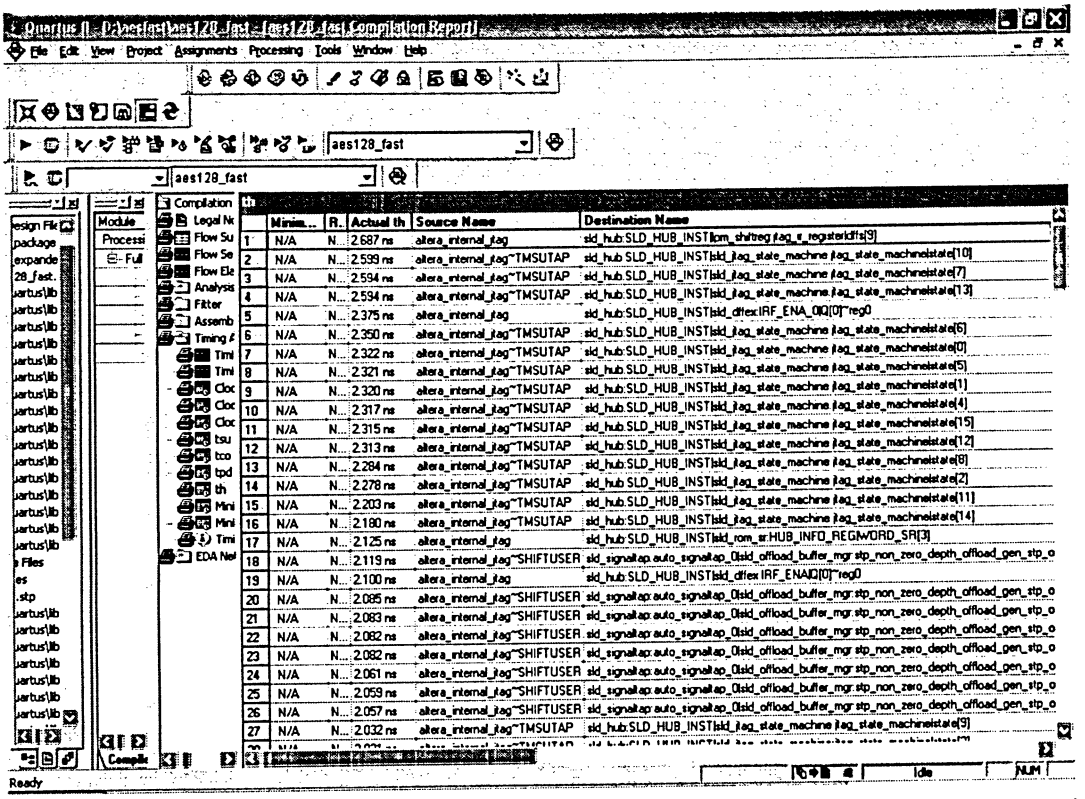

File Edit View Project Assignments Processing Tools Window Help
aes128_fast
aes128_fast
Type
S.
R.
Actual Time
Source Name
Destination Name
1 Clock Setup 'clk' N N 61.72 MHz (period = 16.201 ns) s2_buf[1][4] s3_buf[2][4]
2 Clock Setup 'altera_internal_jtag~TCKUTAP' N N 117.18 MHz (period = 8.534 ns) sld_hub:SLD_HUB_INS sld_hub:SLD_HUB_INS
3 Clock Setup 'altera_internal_jtag~CLKDRUSER' N N 170.33 MHz (period = 5.871 ns) sld_signaltap:auto_sign... sld_signaltap:auto_sign...
4 Worst-case tsu N N 15.398 ns mode s3_buf[2][4]
5 Worst-case tco N N 11.670 ns data_out[18]~reg0 data_out[18]
6 Worst-case tpd N N 2.758 ns altera_internal_jtag~TDO altera_reserved_tdo
7 Worst-case th N N 2.687 ns altera_internal_jtag sld_hub:SLD_HUB_INS
8 Worst-case minimum tco N N 9.098 ns data_out[104]~reg0 data_out[104]
9 Worst-case minimum tpd N N 2.758 ns altera_internal_jtag~TDO altera_reserved_tdo
For Help, press F1
Idle
NUM

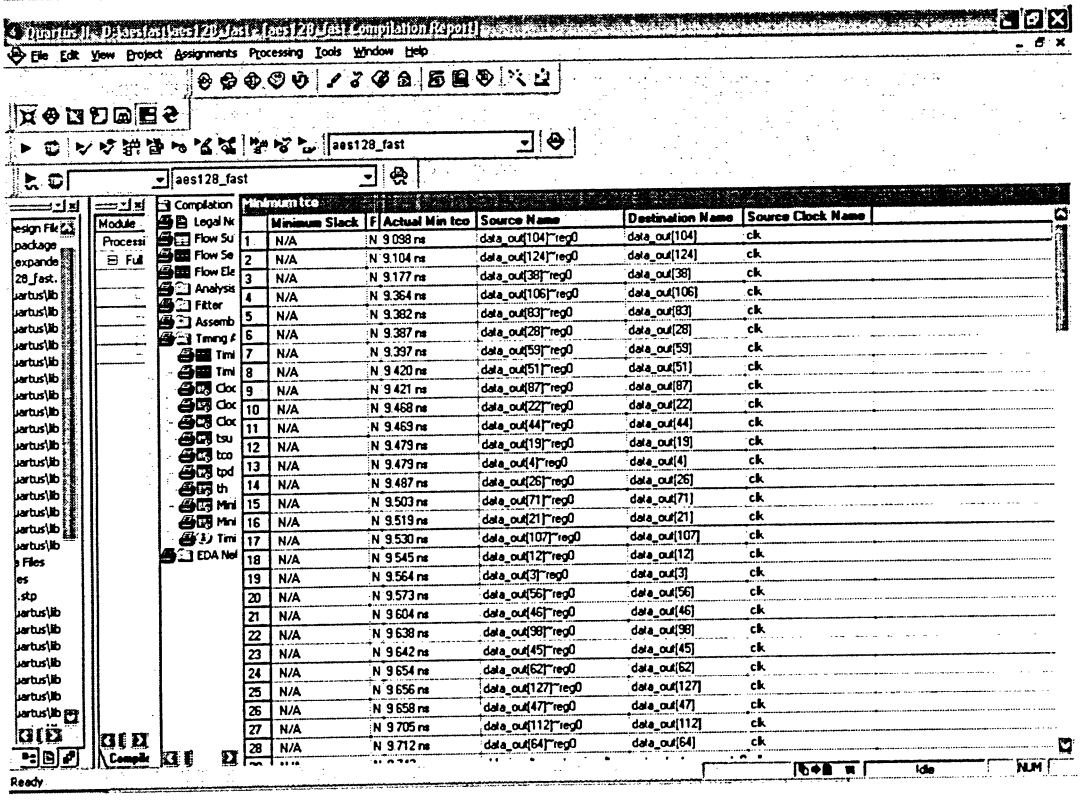

aes128_fast
aes128_fast
Clock Setup: 'clk'
Slack
Actual fmax (period)
Source Name
Destination Name
Source Clo...
Destination Clock Name
1 N/A 61.72 MHz (period = 16.201 ns) s2_buf[1][4] s3_buf[2][4] clk clk
2 N/A 62.79 MHz (period = 15.927 ns) s2_buf[1][5] s3_buf[2][4] clk clk
3 N/A 62.85 MHz (period = 15.910 ns) s2_buf[1][1] s3_buf[2][4] clk clk
4 N/A 63.20 MHz (period = 15.824 ns) s2_buf[1][0] s3_buf[2][4] clk clk
5 N/A 65.48 MHz (period = 15.272 ns) s2_buf[1][2] s1_buf[3][4] clk clk
6 N/A 65.55 MHz (period = 15.255 ns) s2_buf[1][3] s3_buf[2][4] clk clk
7 N/A 65.65 MHz (period = 15.232 ns) s2_buf[1][4] s1_buf[3][4] clk clk
8 N/A 65.79 MHz (period = 15.200 ns) s2_buf[1][6] s3_buf[2][4] clk clk
9 N/A 65.92 MHz (period = 15.170 ns) s2_buf[1][0] s1_buf[3][4] clk clk
10 N/A 65.93 MHz (period = 15.167 ns) s2_buf[1][4] s3_buf[0][4] clk clk
11 N/A 66.02 MHz (period = 15.146 ns) s0_buf[1][5] s3_buf[2][4] clk clk
12 N/A 66.04 MHz (period = 15.142 ns) s0_buf[1][1] s3_buf[2][4] clk clk
13 N/A 66.15 MHz (period = 15.117 ns) s0_buf[1][7] s3_buf[2][4] clk clk
14 N/A 66.41 MHz (period = 15.059 ns) s2_buf[1][4] s3_buf[3][3] clk clk
15 N/A 66.60 MHz (period = 15.016 ns) key_expander:expand_ke... key_expander:expand_ke... clk clk
16 N/A 66.75 MHz (period = 14.981 ns) s0_buf[1][3] s3_buf[2][4] clk clk
17 N/A 66.96 MHz (period = 14.935 ns) s2_buf[1][4] s3_buf[2][3] clk clk
18 N/A 66.98 MHz (period = 14.930 ns) s2_buf[1][1] s1_buf[3][4] clk clk
19 N/A 67.02 MHz (period = 14.922 ns) s3_buf[1][0] s0_buf[2][6] clk clk
20 N/A 67.02 MHz (period = 14.920 ns) s2_buf[1][4] s3_buf[3][4] clk clk
21 N/A 67.05 MHz (period = 14.915 ns) key_expander:expand_ke... key_expander:expand_ke... clk clk
22 N/A 67.13 MHz (period = 14.896 ns) key_expander:expand_ke... new_key0_d1[3][1] clk clk
23 N/A 67.15 MHz (period = 14.893 ns) s2_buf[1][5] s3_buf[0][4] clk clk
24 N/A 67.15 MHz (period = 14.891 ns) key_expander:expand_ke... key_expander:expand_ke... clk clk
25 N/A 67.20 MHz (period = 14.880 ns) s2_buf[1][4] s3_buf[1][4] clk clk
26 N/A 67.22 MHz (period = 14.877 ns) key_expander:expand_ke... key_expander:expand_ke... clk clk
27 N/A 67.22 MHz (period = 14.876 ns) s2_buf[1][1] s3_buf[0][4] clk clk
Ready

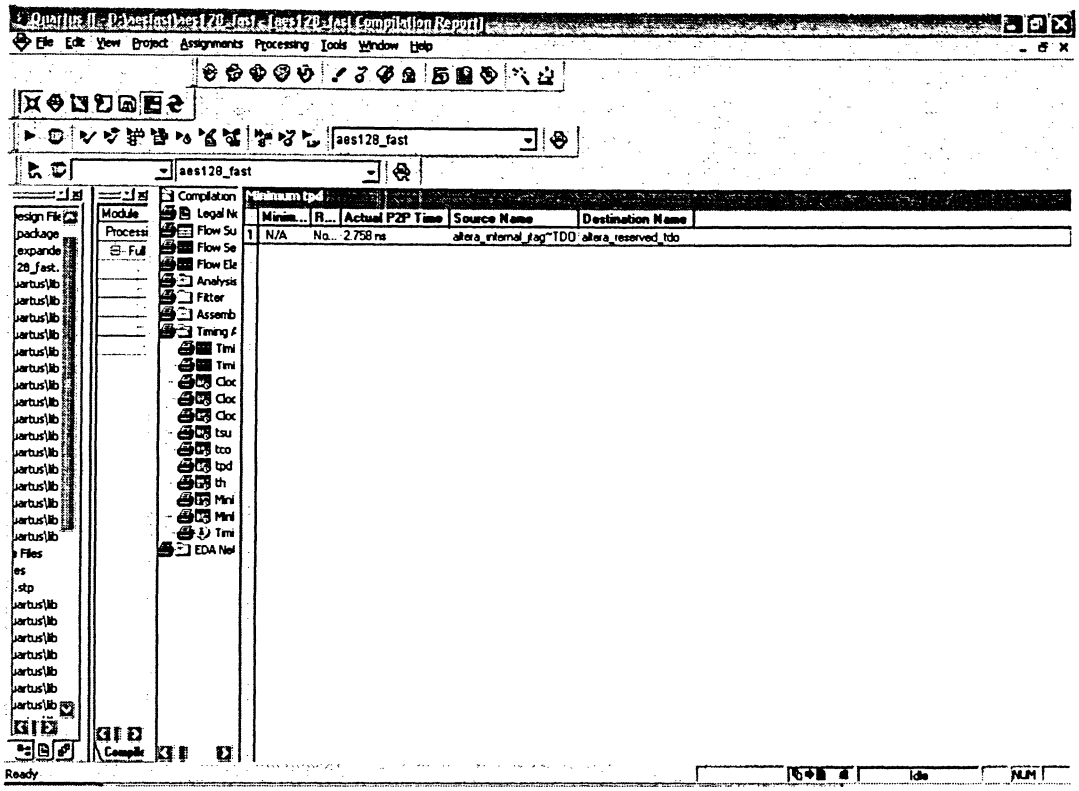

aes128_fast
aes128_fast
Slack
R...
Actual tsu
Source Name
Destination Name
Destination Clock Name
1 N/A No... 15.388 ns mode s3_buf[2][4] clk
2 N/A No... 15.267 ns mode s0_buf[1][1] clk
3 N/A No... 15.096 ns mode s2_buf[0][4] clk
4 N/A No... 15.085 ns mode s1_buf[1][1] clk
5 N/A No... 14.965 ns mode s0_buf[2][6] clk
6 N/A No... 14.927 ns mode data_out[113]~reg0 clk
7 N/A No... 14.667 ns mode s1_buf[0][5] clk
8 N/A No... 14.651 ns mode s2_buf[1][7] clk
9 N/A No... 14.590 ns mode s0_buf[2][4] clk
10 N/A No... 14.568 ns mode s3_buf[0][5] clk
11 N/A No... 14.556 ns mode s2_buf[0][1] clk
12 N/A No... 14.543 ns mode s2_buf[2][3] clk
13 N/A No... 14.510 ns mode s0_buf[3][6] clk
14 N/A No... 14.496 ns mode s1_buf[3][4] clk
15 N/A No... 14.492 ns mode key_expander:expand_keyiw_i_nk2[3][1] clk
16 N/A No... 14.477 ns mode s1_buf[0][4] clk
17 N/A No... 14.467 ns mode s0_buf[1][0] clk
18 N/A No... 14.462 ns mode s0_buf[1][4] clk
19 N/A No... 14.435 ns mode s0_buf[1][2] clk
20 N/A No... 14.404 ns mode s0_buf[0][7] clk
21 N/A No... 14.397 ns mode s3_buf[0][7] clk
22 N/A No... 14.395 ns mode s2_buf[1][1] clk
23 N/A No... 14.386 ns mode s3_buf[3][4] clk
24 N/A No... 14.373 ns mode s2_buf[3][4] clk
25 N/A No... 14.372 ns mode new_key0_d1[3][1] clk
26 N/A No... 14.371 ns mode s2_buf[2][4] clk
27 N/A No... 14.367 ns mode key_expander:expand_keyiw_i_nk0[3][1] clk
28 N/A No... 14.358 ns mode s0_buf[1][7] clk
Ready


Quartus II - D:\aes\fast\aes128_fast - [aes128_fast Compilation Report]
File Edit View Project Assignments Processing Tools Window Help
aes128_fast
aes128_fast
Compilation
Module
Processi
Full
tco
Slack
Required tco
Actual tco
Source Name
Destination Name
Source Clock Na
1 N/A None 11.670 ns data_out[18]~reg0 data_out[18] clk
2 N/A None 11.423 ns data_out[20]~reg0 data_out[20] clk
3 N/A None 11.265 ns data_out[5]~reg0 data_out[5] clk
4 N/A None 11.220 ns data_out[118]~reg0 data_out[118] clk
5 N/A None 11.193 ns data_out[9]~reg0 data_out[9] clk
6 N/A None 11.171 ns data_out[37]~reg0 data_out[37] clk
7 N/A None 11.084 ns data_out[60]~reg0 data_out[60] clk
8 N/A None 11.030 ns data_out[116]~reg0 data_out[116] clk
9 N/A None 10.880 ns data_out[16]~reg0 data_out[16] clk
10 N/A None 10.843 ns data_out[10]~reg0 data_out[10] clk
11 N/A None 10.840 ns data_out[65]~reg0 data_out[65] clk
12 N/A None 10.803 ns data_out[100]~reg0 data_out[100] clk
13 N/A None 10.768 ns data_out[17]~reg0 data_out[17] clk
14 N/A None 10.764 ns data_out[109]~reg0 data_out[109] clk
15 N/A None 10.730 ns data_out[96]~reg0 data_out[96] clk
16 N/A None 10.730 ns data_out[117]~reg0 data_out[117] clk
17 N/A None 10.702 ns data_out[32]~reg0 data_out[32] clk
18 N/A None 10.679 ns data_out[55]~reg0 data_out[55] clk
19 N/A None 10.652 ns data_out[99]~reg0 data_out[99] clk
20 N/A None 10.643 ns data_out[102]~reg0 data_out[102] clk
21 N/A None 10.581 ns data_out[63]~reg0 data_out[63] clk
22 N/A None 10.560 ns data_out[119]~reg0 data_out[119] clk
23 N/A None 10.552 ns data_out[24]~reg0 data_out[24] clk
24 N/A None 10.529 ns data_out[80]~reg0 data_out[80] clk
25 N/A None 10.521 ns data_out[13]~reg0 data_out[13] clk
26 N/A None 10.500 ns data_out[52]~reg0 data_out[52] clk
27 N/A None 10.484 ns data_out[31]~reg0 data_out[31] clk
tco
NUM
Ready

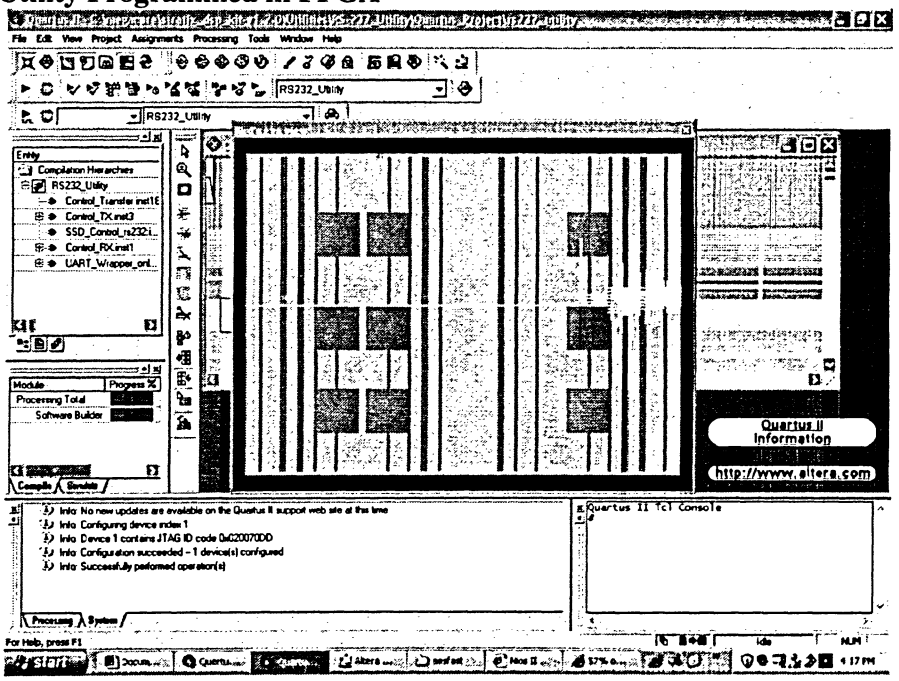

Quartus II - D:\aes\fast\aes128_fast - [aes128_fast Compilation Report]
File Edit View Project Assignments Processing Tools Window Help
aes128_fast
aes128_fast
Compilation
Module
Processi
Full
tpd
Slack
R.
Actual P2P Time
Source Name
Destination Name
1 N/A N... 2.758 ns altera_internal_jtag~TDO altera_reserved_tdo
tpd
NUM
Ready


File Edit View Project Assignments Processing Tools Window Help
aes128_fast
Minimum th
Minim... R. Actual th Source Name Destination Name
1 N/A N... 2.687 ns altera_internal_jtag sld_hub:SLD_HUB_INST|lpm_shiftreg:jtag_ir_register|dffs[3]
2 N/A N... 2.599 ns altera_internal_jtag~TMSUTAP sld_hub:SLD_HUB_INST|sld_jtag_state_machine:jtag_state_machine|state[10]
3 N/A N... 2.594 ns altera_internal_jtag~TMSUTAP sld_hub:SLD_HUB_INST|sld_jtag_state_machine:jtag_state_machine|state[7]
4 N/A N... 2.594 ns altera_internal_jtag~TMSUTAP sld_hub:SLD_HUB_INST|sld_jtag_state_machine:jtag_state_machine|state[13]
5 N/A N... 2.375 ns altera_internal_jtag sld_hub:SLD_HUB_INST|sld_dffex:IRF_ENA_0|Q[0]~reg0
6 N/A N... 2.350 ns altera_internal_jtag~TMSUTAP sld_hub:SLD_HUB_INST|sld_jtag_state_machine:jtag_state_machine|state[6]
7 N/A N... 2.322 ns altera_internal_jtag~TMSUTAP sld_hub:SLD_HUB_INST|sld_jtag_state_machine:jtag_state_machine|state[0]
8 N/A N... 2.321 ns altera_internal_jtag~TMSUTAP sld_hub:SLD_HUB_INST|sld_jtag_state_machine:jtag_state_machine|state[5]
9 N/A N... 2.320 ns altera_internal_jtag~TMSUTAP sld_hub:SLD_HUB_INST|sld_jtag_state_machine:jtag_state_machine|state[1]
10 N/A N... 2.317 ns altera_internal_jtag~TMSUTAP sld_hub:SLD_HUB_INST|sld_jtag_state_machine:jtag_state_machine|state[4]
11 N/A N... 2.315 ns altera_internal_jtag~TMSUTAP sld_hub:SLD_HUB_INST|sld_jtag_state_machine:jtag_state_machine|state[15]
12 N/A N... 2.313 ns altera_internal_jtag~TMSUTAP sld_hub:SLD_HUB_INST|sld_jtag_state_machine:jtag_state_machine|state[12]
13 N/A N... 2.284 ns altera_internal_jtag~TMSUTAP sld_hub:SLD_HUB_INST|sld_jtag_state_machine:jtag_state_machine|state[8]
14 N/A N... 2.278 ns altera_internal_jtag~TMSUTAP sld_hub:SLD_HUB_INST|sld_jtag_state_machine:jtag_state_machine|state[2]
15 N/A N... 2.203 ns altera_internal_jtag~TMSUTAP sld_hub:SLD_HUB_INST|sld_jtag_state_machine:jtag_state_machine|state[11]
16 N/A N... 2.180 ns altera_internal_jtag~TMSUTAP sld_hub:SLD_HUB_INST|sld_jtag_state_machine:jtag_state_machine|state[14]
17 N/A N... 2.125 ns altera_internal_jtag sld_hub:SLD_HUB_INST|sld_rom_sr:HUB_INFO_REG|WORD_SR[3]
18 N/A N... 2.119 ns altera_internal_jtag~SHIFTUSER sld_signaltap:auto_signaltap_0|sld_offload_buffer_mgr:stp_non_zero_depth_offload_gen_stp_o
19 N/A N... 2.100 ns altera_internal_jtag sld_hub:SLD_HUB_INST|sld_dffex:IRF_ENA[0]|Q[0]~reg0
20 N/A N... 2.085 ns altera_internal_jtag~SHIFTUSER sld_signaltap:auto_signaltap_0|sld_offload_buffer_mgr:stp_non_zero_depth_offload_gen_stp_o
21 N/A N... 2.083 ns altera_internal_jtag~SHIFTUSER sld_signaltap:auto_signaltap_0|sld_offload_buffer_mgr:stp_non_zero_depth_offload_gen_stp_o
22 N/A N... 2.082 ns altera_internal_jtag~SHIFTUSER sld_signaltap:auto_signaltap_0|sld_offload_buffer_mgr:stp_non_zero_depth_offload_gen_stp_o
23 N/A N... 2.082 ns altera_internal_jtag~SHIFTUSER sld_signaltap:auto_signaltap_0|sld_offload_buffer_mgr:stp_non_zero_depth_offload_gen_stp_o
24 N/A N... 2.061 ns altera_internal_jtag~SHIFTUSER sld_signaltap:auto_signaltap_0|sld_offload_buffer_mgr:stp_non_zero_depth_offload_gen_stp_o
25 N/A N... 2.059 ns altera_internal_jtag~SHIFTUSER sld_signaltap:auto_signaltap_0|sld_offload_buffer_mgr:stp_non_zero_depth_offload_gen_stp_o
26 N/A N... 2.057 ns altera_internal_jtag~SHIFTUSER sld_signaltap:auto_signaltap_0|sld_offload_buffer_mgr:stp_non_zero_depth_offload_gen_stp_o
27 N/A N... 2.032 ns altera_internal_jtag~TMSUTAP sld_hub:SLD_HUB_INST|sld_jtag_state_machine:jtag_state_machine|state[9]
Idle
NUM
Ready


File Edit View Project Assignments Processing Tools Window Help
aes128_fast
Minimum tco
Minimum Slack F Actual Min tco Source Name Destination Name Source Clock Name
1 N/A N 9.098 ns data_out[104]~reg0 data_out[104] clk
2 N/A N 9.104 ns data_out[124]~reg0 data_out[124] clk
3 N/A N 9.177 ns data_out[38]~reg0 data_out[38] clk
4 N/A N 9.364 ns data_out[106]~reg0 data_out[106] clk
5 N/A N 9.382 ns data_out[83]~reg0 data_out[83] clk
6 N/A N 9.387 ns data_out[28]~reg0 data_out[28] clk
7 N/A N 9.397 ns data_out[59]~reg0 data_out[59] clk
8 N/A N 9.420 ns data_out[51]~reg0 data_out[51] clk
9 N/A N 9.421 ns data_out[87]~reg0 data_out[87] clk
10 N/A N 9.468 ns data_out[22]~reg0 data_out[22] clk
11 N/A N 9.469 ns data_out[44]~reg0 data_out[44] clk
12 N/A N 9.479 ns data_out[19]~reg0 data_out[19] clk
13 N/A N 9.479 ns data_out[4]~reg0 data_out[4] clk
14 N/A N 9.487 ns data_out[26]~reg0 data_out[26] clk
15 N/A N 9.503 ns data_out[71]~reg0 data_out[71] clk
16 N/A N 9.519 ns data_out[21]~reg0 data_out[21] clk
17 N/A N 9.530 ns data_out[107]~reg0 data_out[107] clk
18 N/A N 9.545 ns data_out[12]~reg0 data_out[12] clk
19 N/A N 9.564 ns data_out[3]~reg0 data_out[3] clk
20 N/A N 9.573 ns data_out[56]~reg0 data_out[56] clk
21 N/A N 9.604 ns data_out[46]~reg0 data_out[46] clk
22 N/A N 9.638 ns data_out[98]~reg0 data_out[98] clk
23 N/A N 9.642 ns data_out[45]~reg0 data_out[45] clk
24 N/A N 9.654 ns data_out[62]~reg0 data_out[62] clk
25 N/A N 9.656 ns data_out[127]~reg0 data_out[127] clk
26 N/A N 9.658 ns data_out[47]~reg0 data_out[47] clk
27 N/A N 9.705 ns data_out[112]~reg0 data_out[112] clk
28 N/A N 9.712 ns data_out[64]~reg0 data_out[64] clk
Idle
NUM
Ready



RS232 Utility Programmed in FPGA



Appendix B

Test data

Input and Output Text from the AES Core after an Encryption Process followed by a Decryption Process

Original Data (Plain Text)	Original Data (Plain Text)	Decrypted Data (Equal to Original Plain Text)
0x7e7ffc5b	0x811098a8	0x7e7ffc5b
0xfefe67ea	0x3e7e5fff	0xfefe67ea
0xf7a7ec9b	0x4aaa0bd8	0xf7a7ec9b
0xb7fdfd74	0x43a22e06	0xb7fdfd74

Bibliography

[1] Daniel W. Engels, "RFID: The Technical Reality,"Auto-ID Labs Massachusetts Institute of Technology, Washington, D.C, 21 June 2004

[2] Klaus Finkenzeller, "RFID Handbook Fundamentals and Applications in Contact less Smart Cards and Identification," Second Edition, Wiley Publication.

[3] Ari Juels, "RFID Security and Privacy: A Research Survey," IEEE Journal on selected areas in Communication, Vol. 24, No. 2, February 2006.

[4] EPCglobal Inc. (2005, May) EPC generation 1 tag data. Standards version 1.1 Rev. 1.27. [Online]. Available: http://www.epcglobalinc.Org

[5] Roger Smith, "RFID: A Brief Technology Analysis," CTO Network Library, 2005.

[6] Intel Research, Seattle. (http://seattleweb.intel-research.net/projects/activity/)

[7] Manfred Aigner, Martin Feldhofer, "Secure Symmetric Authentication for RFID Tags," IAIK Institute for Applied Information Processing and Communication, Graz February, 2005

[8]M. Feldhofer, S. Dominikus, and J. Wolkerstorfer, "Strong Authentication for RFID Systems using the AES Algorithm," *In Conference Proceedings of Cryptographic Hardware and Embedded Systems*, Pages 357-370.Spring 2004.

[9] M. Feldhofer, "An Authentication Protocol in a Security Layer for RFID Smart Tags," *12th IEEE Mediterranean Electrotechnical Conference* – MELECON 2004, IEEE Proceedings. Pages 759-762, May 2004

[10] National Institute of Standards and Technology (NIST). FIPS-197: "Advanced Encryption Standard (AES)" November 2001. Available online at http://www.itl.nist.gov/fipspubs/

[11] International Organization for Standardization (ISO). ISO/IEC 9798-2: *I*nformation Technology – Security Techniques – Entity authentication mechanisms – Part 2: "Mechanisms using symmetric encipherment algorithms,"1993.

[12] International Organization for Standardization (ISO). ISO/IEC 18000-3: Information Technology AIDC Techniques – RFID for Item Management, 2003

[13] M. Jakobsson and S. Wetzel, "Security weaknesses in Bluetooth," in the Cryptographer's Track at RSA, D. Naccache, Ed. New York: Springer-Verlag, 2001, vol. 2020, Lecture Notes in Computer Science, pp. 176–191.

[14] J. Westhues, "Hacking the prox card," in *RFID: Applications, Security, and Privacy*, S. Garfinkel and B. Rosenberg, Eds. Reading, MA: Addison- Wesley, 2005, pp. 291– 300.

[15] S. Bono, M. Green, A. Stubblefield, A. Juels, A. Rubin, and M. Szydlo, "Security Analysis of a cryptographically-enabled RFID device," in Proc. 14th USENIX Security Symposium, P. McDaniel, Ed., 2005, pp. 1–16.Available online: www.rfidanalysis.org.

[16] United States Food and Drug Administration. Combating counterfeit drugs: A report of the food and drug administration [Online]. Available:http://www.fda.gov/oc/initiatives/counterfeit/report02_04.html

[17] Texas Instruments and VeriSign, Inc., "securing the pharmaceutical supply chain with RFID and public-key infrastructure technologies," Whitepaper [Online]. Available: http://www.ti.com/rfid/docs/customer/ eped-form.shtml

[18] Texas Instruments Gen 2 Inlay Data Sheet (2005) [Online]. Available: http://www.ti.com/rfid/docs/manuals/pdfSpecs/epc_inlay.pdf

[19] G. Avoine and P. Oechslin, "RFID traceability: A multilayer problem," in *Proc. Financial Cryptography*, A. Patrick and M. Yung, Eds. New York: Springer-Verlag, 2005, vol. 3570, Lecture Notes in Computer Science, pp. 125–140.

[20] S. A.Weis, "Radio-Frequency Identification Security and Privacy," M.S.Thesis, MIT, Cambridge, MA, June, 2003.

[21] Ari Juels, "RFID Security and Privacy: A Research Survey," IEEE Journal on selected areas in Communications," Vol. 24, No. 2, February 2006.

[22] S. E. Sarma, S. A.Weis, and D.W. Engels, "RFID systems, security and privacy Implications," AutoID Center, MIT, Cambridge, MA, Tech. Rep. MIT, AUTOID- WH-014, 2002.

[23] N. Good, J. Han, E. Miles, D. Molnar, D.Mulligan, L. Quilter, J. Urban, and D. Wagner, "Radio frequency identification and privacy with information goods," in *Proc. Workshop on Privacy in the Electronic Society—WPES*, S. De Capitani di Vimercati and P. Syverson, Eds, 2004, pp. 41–42.

[24] A. Juels, "Minimalist cryptography for low-cost RFID tags," in Proc.4th International Conference of Security Communication and Network, 2004.

[25] A. Juels and R. Pappu, "Squealing Euros: Privacy protection in RFID-enabled banknotes," in *Proc. Financial Cryptography*, R.Wright, Ed. New York: Springer-Verlag, 2003, vol. 2742, Lecture Notes in Computer Science, pp. 103–121.

[26] S. Bono et al., "Security Analysis of a Cryptographically- Enabled RFID Device," Usenix Security, P.McDaniel, ed., Usenix Assoc., 2005.

[27] S.L.Garfinkel, Ari Juels and Ravi Pappu, "RFID Privacy: An overview of problems and proposed solutions," IEEE transaction on Security and Privacy, 1540-7993/05/ 2005.

[28] A. Juels and J. Brainard, "Soft Blocking: Flexible Blocker Tags on the Cheap," Workshop on Privacy in the Electronic Society (WPES 04), ACM Press, 2004.

[29] Kris Gaj and Pawel Chodowiec, "Comparison of the hardware performance of the AES candidates using reconfigurable hardware", George Mason University, 2005.

[30] Fred Mohamadi, "A miniature reader/active tag streamlines supply chain management ," www.rfdesign.com, September 2004

[31] WJ Communication Inc., "A Compact RFID reader platform for UHF and Microwave application," Microwave Journal, April 2004

[32] John Parkinson, "RFID and the Consumer Understanding Their Mindset" R*esults of a Survey by Capgemini for the National Retail Federation,* FTC Workshop on RFID, Washington DC, June 21, 2004

[33] R. Fletcher, O. Omojola, E. Boyden & N. Gershenfeld, "Reconfigurable Agile Tag Reader Technologies for Combined EAS and RFID Capability," Massachusetts Institute of Technology, Media Laboratory, Cambridge, MA

[34] Ken Fishkin, Intel-Research, "RFID for Healthcare: Some Current and Anticipated Uses," FTC Workshop – Fishkin, June 2004

[35] Christopher Boone, "RFID: The Next Big Thing?" FTC RFID Workshop, June 2004

[36] Paul Rudolf, "Combating Counterfeit Drugs: Use of RFID," U.S. Food and Drug Administration, May 2005.

[37] Sunny Dzik, Blood Transfusion Service, "Case Study: RFID in Action – the Massachusetts General Hospital START project," Study conducted in Massachusetts General Hospital, 2004

[38] William Allen Texas Instruments, Inc. TI-RFID Systems, "Current and Anticipated Uses of RFID Technology," *Federal Trade Commission Workshop on RFID*

[39] Cesare Alippi, Giovanni Vanini, "An Application-Level Methodology to Guide the Design of Intelligent-Processing, Power-Aware Passive RFID," 2005 IEEE.

[40] Matthai Philipose and Joshua R.Smith, Bing Jiang, Alexander Mamishev, and Sumit Roy, Kishore Sundara-Rajan, "Battery-Free Wireless Identification and Sensing," IEEE Pervasive computing, 1536-1268/05

[41] Urban Bilstrup, Per-Arne Wiberg, "An Architecture Comparison between a Wireless Sensor Network and an Active RFID System," Proceedings of the 29th Annual IEEE International Conference on Local Computer Networks, (LCN'04) 0742-1303/04

[42] Le-Pong Chin and Chia-Lin Wu, "The Role of Electronic Container Seal (E-Seal) With RFID Technology in the Container Security Initiatives," Proceedings of the 2004 International Conference on MEMS, NANO and Smart Systems, 0-7695-2189-4/04

[43] S. Basat, K. Lim, I. Kim, M.M. Tentzeris, J. Laskar, "Design and Development of a Miniaturized Embedded UHF RFID Tag for Automotive Tire Applications," IEEE Electronic Components and Technology Conference, 0-7803-8906-9 /2005

[44] Vince Stanford, "Pervasive Computing Goes the Last Hundred Feet with RFID Systems," IEEE Pervasive computing Magazine, IEEE CS and IEEE Comsoc, 1536- 1268/03/2003

[45] Vince Stanford, "Using Pervasive Computing to Deliver Elder Care," IEEE Pervasive computing, 536-1268/02/2002

[46] Xingxin (Grace) Gao, Zhe (Alex) Xiang, Hao Wang, Jun Shen, Jian Huang, Song Song, "An Approach to Security and Privacy of RFID System for Supply Chain," Proceedings of the IEEE International Conference on E-Commerce Technology for Dynamic E-Business, (CEC-East'04) 0-7695-2206-8/2004

[47] http://sourceforge.net/projects/libaes/ , http://xyssl.org/code/source/aes/ & Paulo Barreto's public domain C implementation of AES

[48] Ron Weinstein, "RFID: A Technical Overview and Its Application to the Enterprise," Published by the IEEE Computer Society, June 2005