Compromising Emissions from a High Speed Cryptographic Embedded System

A Thesis Submitted in Fulfillment of the
Requirements for the Degree of
Master of Engineering
(Electronic Engineering)

By
Damien O'Rourke
B.Eng., MIEEE

School of Electronic Engineering
Faculty of Engineering and Design
Dublin City University

Research Supervisor
Prof. Charles McCorkell
Msc, PhD, CEng, FIEI, FIEE

September 2003

Declaration

I hereby certify that this material, which I now submit for assessment on the programme of study leading to the award of Master of Engineering is entirely my own work and has not been taken from the work of others save and to the extent that such work has been cited and acknowledged within the text of my work.

Signed: Daniel O'Loughlin Date: 01-10-03

Acknowledgments

I would like to thank my research supervisor and colleague Prof. Charles McCorkell for all his support and advice over the duration of this thesis. Without his guidance, encouragement and patience, this work would have invariably been a lot more difficult.

I would also like to express a special thanks to Prof. Michael Ryan for sharing his invaluable knowledge in the area of cryptography and for his complete patience in helping me whenever it was needed.

I am grateful to Frank Flanagan and Richard Rooney for all their help throughout this research and also to Dr. Liam Barry for making me an "honorary" member of his lab and for including me in all the lab events.

I would especially like to thank my parents Margaret and Terry for supporting me in every decision I have ever made and for always being there.

To my dear friend and girlfriend Claire. I couldn't have done this without you.

I am very grateful to the Mr. Conor Maguire, Mr. Robert Clare, Mr. Liam Meany, Mr. John Whelan, Mr. Jim Dowling, Mr. Liam Sweeney, Mr. Paul Wogan, Mr. Billy Roarty and Dr. Pascal Landais for assisting me in various aspects of my work in Dublin City University. Thank you all for your time and friendship.

I would like to thank my friends and colleagues here at the University: Eoin, Clauss, Paul, Frank, Brendan, Prince, Ola, Antonia and Ashling. Your friendship and support has been invaluable, thank you all.

I would like to thank all my friends for being there and for lending me an ear when things got tough. I would especially like to thank Robert Clarke and Colin Smith who being the nerds they are, kept me highly motivated.

Last but certainly not least, I would like to thank God for blessing me with such a beautiful family and friends, and for helping me to come this far.

Dedication

To Mam, Dad and Claire - this is for you.

Abstract

Compromising Emissions from a High Speed Cryptographic Embedded System

Damien O'Rourke, *B.Eng.*, *MIEEE*

Specific hardware implementations of cryptographic algorithms have been subject to a number of "side channel" attacks of late. A side channel is any information bearing emission that results from the physical implementation of a cryptographic algorithm. Smartcard realisations have been shown to be particularly vulnerable to these attacks. Other more complex embedded cryptographic systems may also be vulnerable, and each new design needs to be tested.

The vulnerability of a recently developed high speed cryptographic accelerator is examined. The purpose of this examination is not only to verify the integrity of the device, but also to allow its designers to make a determination of its level of conformance with any standard that they may wish to comply with.

A number of attacks were reviewed initially and two were chosen for examination and implementation - Power Analysis and Electromagnetic Analysis. These particular attacks appeared to offer the greatest threat to this particular system. Experimental techniques were devised to implement these attacks and a simulation and micrcontroller emulation were setup to ensure these techniques were sound.

Each experimental setup was successful in attacking the simulated data and the micrcontroller circuit. The significance of this was twofold in that it verified the integrity of the setup and proved that a real threat existed. However, the attacks on the cryptographic accelerator failed in all cases to reveal any significant information. Although this is considered a positive result, it does not prove the integrity of the device as it may be possible for an adversary with more resources to successfully attack the board. It does however increase the level of confidence in this particular product and acts as a stepping stone towards conformance of cryptographic standards.

The experimental procedures developed can also be used by designers wishing to test the vulnerability of their own products to these attacks.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

The idea of communications interception is not new. For as long as human beings have been communicating, some unintended recipient has been listening. With the advent of technology, communication methods have improved dramatically, and ways have been devised to make it increasingly difficult for unauthorised parties to intercept and analyse the information communicated. Unfortunately, using the same technology, the methods of interception have also greatly improved.

Secure communications can be achieved to an extent using a technique known as *Cryptography* - the art of secret writing. Cryptography has been around for centuries in one form or another and has received new life with the extraordinary computing power available in todays microprocessors. During the two world wars, cryptography was a highly classified subject and little was known about it in the public domain. With the advancement of technology however, public domain research flourished, and the first public algorithms were developed. Not surprisingly, with these algorithms came reported attacks on them.

Until recently most of the attacks were aimed at the algorithm itself. Mathematical weakness's were exploited to allow the secret information to be revealed. However as the theoretical integrity of the algorithms evolved, direct attacks became much less effective. Attackers looked for alternative ways to compromise the security of these algorithms and inevitably found them - the so called *side channels*.

A "side channel" is any information bearing emission[1] that results from the physical implementation of a cryptographic algorithm. Due to the fact that these emissions are information bearing, they are referred to as *compromising emissions*.

This thesis aims to test the vulnerability of a certain high speed *cryptographic device* to some of these compromising emissions (specifically the first and last attacks outlined in section 4.11 of the Federal Information Processing Standard (FIPS) 140-2, "Security Requirements For Cryptographic Modules" [1]). A cryptographic device (also known as a cryptoprocessor) is any electronic or mechanical[2] device that is used to keep information secure. Published research would suggest that side channels have been used primarily to attack smart cards and other low power, minimal circuitry devices. The high speed cryptographic device tested here (herein referred to as the Device Under Test or DUT) is far more complicated than any smart card. It has it's own purpose built Application Specific Integrated Circuit (ASIC), and also contains at least one countermeasure to these attacks. It will be seen that the attacks were far more difficult to implement in this case.

1.1 Cryptography

The idea of cryptography is to convert information to a form that will be unintelligible to an unintended recipient. It accomplishes this using a cryptographic algorithm known as a *cipher*, and an object known as a *cryptographic key*. Figures 1.1 and 1.2 show examples of the two classes of cryptographic systems in use today. The information to be converted is known as *plaintext* and the converted information is known as *ciphertext*. The process of converting plaintext to ciphertext is known as *encryption* and the process of converting back from ciphertext to plaintext is known as *decryption*.

[1] The US military refers to it as an "emanation" however the two words will be used interchangeably throughout this thesis.

[2] Today they are built for practical applications using electronics only, however during World War II, they were entirely mechanical.

The cryptographic key may (figure 1.1) or may not (figure 1.2) be the same for both encryption and decryption depending on the type of algorithm used. A dedicated circuit board used to perform encryption and decryption is known as a *cryptoprocessor* which can be part of an overall system known as a *cryptosystem*[3].

There are two main types of cryptography in use today - *symmetric* or *secret key* cryptography and *asymmetric* or *public key* cryptography. Symmetric key cryptography is the oldest type whereas asymmetric cryptography is only being used publicly since the late 1970's[4]. Asymmetric cryptography was a major milestone in the search for a perfect encryption scheme.

1.1.1 Secret Key Cryptography

As mentioned, secret key cryptography is by far the oldest of the two types and goes back to Egyptian times. It involves the use of only one key which is used for both encryption and decryption (hence the name symmetric key cryptography). Figure 1.1 depicts this idea. It is necessary for security purposes that the secret key never be revealed.

Figure 1.1: Secret key encryption.

To accomplish encryption, most secret key algorithms use two main techniques known

[3]Sometimes the two terms are used interchangeably.
[4]It is claimed by some that government agencies knew about asymmetric cryptography before this.

as *substitution* and *permutation*. Substitution is simply a mapping of one value to another whereas permutation is a reordering of the bit positions for each of the inputs. These techniques are used a number of times in iterations called *rounds*. Generally, the more rounds there are, the more secure the algorithm. A non-linearity is also introduced into the encryption so that decryption will be computationally infeasible[5] without the secret key. This is achieved with the use of *S-boxes* which are basically non-linear substitution tables where either the output is smaller than the input or vice versa. See [2] for more details.

For the past 25 years, the main standard for encrypting data was a symmetric algorithm known as the *Data Encryption Standard* (DES)[6]. DES is a 64 bit *block cipher* which means that it encrypts data 64 bits at a time. This is contrasted to a *stream cipher* in which only one bit at a time (or sometimes small groups of bits such as a byte) is encrypted.

DES was the result of a research project set up by International Business Machines (IBM) corporation in the late 1960's which resulted in a cipher known as LUCIFER. In the early 1970's it was decided to commercialise LUCIFER and a number of significant changes were introduced. IBM was not the only one involved in these changes as they sought technical advice from the National Security Agency (NSA) [3] (other outside consultants were involved but it is likely that the NSA were the major contributors from a technical point of view). The altered version of LUCIFER was put forward as a proposal for the new national encryption standard requested by the National Bureau of Standards (NBS)[7]. It was finally adopted in 1977 as the Data Encryption Standard - DES.

Some of the changes made to LUCIFER have been the subject of much controversy

[5]This means that it costs more to implement the attack than the information is worth.
[6]This is now being replaced by a new standard known as the *Advanced Encryption Standard* (AES).
[7]Now known as the National Institute of Standards and Technology (NIST).

even to the present day. The most notable of these was the key size. LUCIFER used a key size of 128 bits however this was reduced to 56 bits for DES. Even though DES actually accepts a 64 bit key as input, the remaining eight bits are used for parity checking and have no effect on DES's security. Outsiders were convinced that the 56 bit key was an easy target for a brute force[8] attack due to its extremely small size. The need for the parity checking scheme was also questioned without satisfying answers.

Another controversial issue was that the S-boxes used were designed under classified conditions and no reasons for their particular design were ever given. This led people to assume that the NSA had introduced a "trapdoor" through which they could decrypt any data encrypted by DES even without knowledge of the key. One startling discovery was that the S-boxes appeared to be secure against an attack known as *Differential Cryptanalysis* [4] which was only publicly discovered by Biham and Shamir in 1990. This suggests that the NSA were aware of this attack in 1977; 13 years earlier! A detailed description of DES is given in appendix A and is required for an understanding of the next chapter.

DES of course isn't the only symmetric cipher. There are many others, each with varying levels of complexity. Such ciphers include: IDEA, RC4, RC5, RC6 and the new Advanced Encryption Standard (AES) known as Rijndael. These ciphers are of no concern in this thesis as the DUT wasn't running them. However AES has been under intense scrutiny with regard to side channel attacks because it must be able to withstand them if it is to be the next cryptographic standard. For more information on the first three of these ciphers, the interested reader is referred to [2] and for Rijndael see [5].

One of the main problems with secret key cryptography is *key distribution*. For this form of cryptography to work, both parties must have a copy of the secret key. This

[8] See section 1.2

would have to be communicated over some secure channel which, unfortunately, is not that easy to achieve. As will be seen in the next section however, public key cryptography provides a solution to this.

1.1.2 Public Key Cryptography

Public key cryptography was considered a major achievement in the field, as it offered an excellent solution for the old time problem of key distribution. It was invented by Whitfield Diffie and Martin Hellman in 1976[9] and was such a milestone, that the term *modern cryptography* is often used to refer to cryptography after this date.

Diffie and Hellman produced the first public key algorithm [6]. Unfortunately, it was only useful for key distribution. An algorithm was needed that would allow both encryption and digital signatures. In 1977, Ron Rivest, Adi Shamir and Leonard Adleman fulfilled this requirement with the algorithm known as RSA (an anagram of the first letters from each of the inventors surnames), the first fully fledged public key algorithm. It was, and is still is, the most popular public key algorithm available.

Public key cryptography uses a different technique than secret key cryptography. Instead of using only one cryptographic key, it uses two keys - a *private key*[10] and a *public key*. The private key is kept private and the public key is kept in a public database where anyone can access it. To accomplish this, a variation of a function known as a *one-way function* is used. This is a function that is easy to calculate one way, but extremely difficult (ideally impossible) to reverse. Encryption using such a function would be easy, but decryption would be impossible. Of course an ideal one-way function is of no use because this would then mean that the intended recipient would be unable to decipher the message. Instead, the variation used, allows decryption only if a

[9] It was also independently invented by Ralph Merkle [2].

[10] The term "private key" is generally used instead of "secret key" to distinguish it from the key used in symmetric key cryptography (of which there is only one). Both the secret key in symmetric cryptography and the private key in asymmetric cryptography must not be revealed to anyone however.

certain parameter is known (i.e. the private key). This is known as a *trapdoor one-way function*. See [2, 7, 8] for more details.

A secure communication or transaction takes place by encrypting intended data with the public key and then transferring the information to the intended recipient. The recipient will then decrypt the data with the private key that is paired with public key. This idea is depicted in figure 1.2 where the public key is given the symbol K_u and the private key K_r. The keys always come in pairs and are thus referred to as a *key pair*.

Figure 1.2: Public key encryption.

There have been quite a few algorithms proposed to implement the public key scheme. However, due to either insecurities or impracticalities, most have been rejected. Of the ones that have been accepted, most are good for either encryption or digital signatures, but not both. Only three are considered to work well for both methods and only one of these three has been widely excepted. The three are RSA, ElGamal and Rabin, with RSA being the most widely accepted. RSA has not yet been proven to be secure, however it gets it's wide acceptance from the fact that nobody has succeeded in breaking it. Details on all three ciphers can be found in [2] and an overview of RSA is given in appendix C.

A lot of public key algorithms (including RSA) use concepts from an area of mathe-

matics known as *Number Theory*[11]. One of the main ideas in number theory is *modular arithmetic* and specifically, *modular exponentiation*. Modular exponentiation is used by many public key algorithms to perform encryption and decryption. It is similar to ordinary exponentiation except that the result is not the value of the exponentiation itself, but of the remainder when this value is divided by a number called the modulus.

Considering RSA specifically, the equation to calculate the ciphertext from the plaintext is

$$C = M^e \bmod n \tag{1.1}$$

where C is the ciphertext, M is the plaintext and the set $\{e, n\}$ constitute the public key, K_u.

The original message M can be reconstructed by decryption which is of a similar form,

$$C^d \bmod n = M^{ed} \bmod n = M \bmod n \tag{1.2}$$

where the set $\{d, n\}$ constitute the private key K_r, and the other variables are as before.

It might appear rather strange that $M^{ed} \bmod n = M \bmod n$ however, there are strict rules that govern the choice of e, d and n which must be adhered to in order for this equality to be true (an explanation is given in appendix C where an overview of RSA is given).

The problem with modular exponentiation is that it is computationally intensive and can severely affect the performance of modern e-commerce servers. A solution to the problem is to assign the encryption to a dedicated high speed cryptoprocessor such as the one being reviewed here.

[11] An overview of number theory is given in appendix B.

1.2 Security of cryptosystems

The following is taken from [9]:

> "A secure coprocessor is a general-purpose computing environment that withstands physical attacks and logical attacks. The device must run the programs that it is supposed to, unaltered. You must be able to (remotely) distinguish between the real device and application, and a clever impersonator. The coprocessor must remain secure even if adversaries carry out destructive analysis of one or more devices ... You need a device you can trust even though you can't control its environment."

Traditionally, cryptographic systems have been represented only as mathematical models. Figure 1.3 shows an example of one of these models. In this scenario, an encrypted communication takes place by encrypting some plaintext with a particular cipher, and transferring it through and insecure channel to which some intruder may be connected. On receiving this information, the recipient decrypts it in the usual manner. It is assumed in this case that the security of the entire cryptosystem rests with the particular algorithm used, and that the eavesdropper will only have some plaintext-ciphertext pairs to work with[12].

Algorithms in cryptography are designed to keep information secure. An algorithm is considered secure if there is no attack that can reveal the secret information with less than 2^N tries where N is the number of bits in the cryptographic key (in other words, the security of a cryptosystem should depend on the cryptographic key used and not on the algorithm itself).

An attack where every possible key is tried until the correct one is found is known as a *Brute force attack*. As there are 2^N possible values (known as the *key space*) for an N

[12] A knowledge of the particular algorithm is always assumed. This is known as Kerckhoff's law.

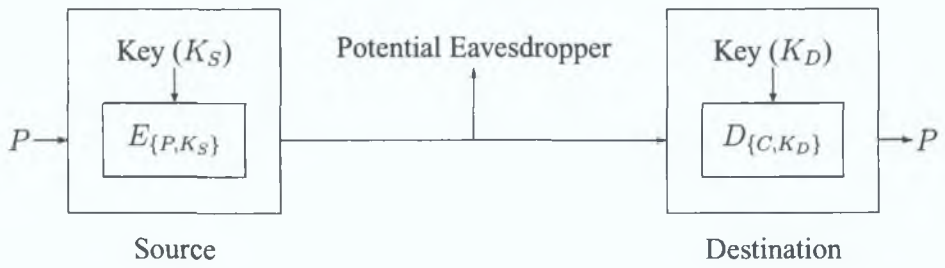


Figure 1.3: Classical model of a cryptographic communication system.

bit number, then generally a brute force attack will take 2^N attempts to find the key[13]. A cryptographic cipher is said to have been *broken* if it is vulnerable to attacks with less effort than a brute force attack (even if the attack is not practical).

There is only one algorithm that has been proven to be completely secure[14] (if used correctly) and is known as the *one time pad*. This is a symmetric cipher where there are as many key bits as plaintext bits. It uses a different key for every encryption and the key set is truly random[15]

Each bit of the plaintext is encrypted with each bit of the key using the exclusive-OR operation ($\oplus$). As the key is truly random, this means that the ciphertext produced will also be truly random and is therefore equally likely to correspond to any plaintext message of the same size as the original. It's security rests on the fact that these random keys are never used more than once. Of course, the person decrypting the message would need to know the keys, and the problem of how best to provide a copy of them is still an issue today. This is the case even with public key cryptography as if this is used to transfer the keys, then the one time pad is only as strong as the public key system used. The one time pad is not a very practical system and is rarely used in real world applications.

[13]On average however, it will only take half this or 2^{N-1} tries.
[14]Even against brute force attacks.
[15]This basically means that there is no way of knowing one key from any combination of the others.

Despite the fact that all practical ciphers are theoretically vulnerable to a brute force attack, it is generally impossible to successfully implement brute force attacks due to the sheer size of the keys. Other attacks which aim to reduce this level of difficulty, by manipulating vulnerabilities in the algorithm itself, generally come up against the problem that most of the algorithms have a very strong theoretical base. However, researchers have demonstrated ways in which a cryptosystem can be compromised without having to attack the algorithm itself [10, 11].

Every cryptographic cipher has to be implemented in one form or another in hardware. Due to the fundamental laws of physics governing this hardware, emissions will be produced that may or may not reveal information about the operations it is running. These emissions can take different forms such as electromagnetic radiation, power fluctuations or timing fluctuations. Some can even be induced by an attacker using a technique known as *fault induction*. As stated already, any emission that can be used to reveal secret information is known as a compromising emission. These emissions have proven to be a serious threat to the security of cryptosystems and are discussed in more detail next.

1.2.1 Compromising Emissions

Compromising Emissions are normally referred to as Compromising Emanations in military and NSA documents. One definition is as follows and is taken from [12]:

> "**Compromising Emanations**: Unintentional signals that, if intercepted and analyzed,would disclose the information transmitted, received, handled, or otherwise processed by information systems equipment".

These signals can be conducted, radiated, reactively coupled, exposed through power and timing fluctuations or even acoustically transmitted. In fact, any way a signal

can be intercepted beyond its intended channels would be considered compromising (assuming that the information is of a sensitive nature).

Power and timing fluctuations are different from the rest in the sense that power and time are resources required by the implementation circuitry to perform its functions. The others are only by-products of its actual operation, and are not required resources. For example, it is not necessary that the device emit electromagnetic radiation in order to operate (assuming it does not use some form of wireless communication) however, it is necessary that the device receive varying amounts of power and time. Reduction of the radiated emissions will not hinder the device in any way, and may even improve it's operation in some instances. On the other hand, reduction of power consumption below its required level will ultimately make the device inoperable.

Whether the signal is a "by-product" emission or a power fluctuation, it is necessary to know how the signal can move from one point to another within a system or between two systems. There are three main physical mechanisms by which a signal can propagate from its original source. These may act together or they may act independently and their effects will be different depending on the systems themselves and on the distance from the source.

1. **Conduction**- Conducted propagations are generated by actual physical connections either within a system or from one system to another or to its power supply. These are most effective at frequencies below 30 MHz. Above this frequency substantial attenuation takes place and it becomes the least dominant mechanism.

2. **Reactive Coupling**- This can exist within a single system or between two systems. There are a number of factors which affect the amount of reactive coupling and these include, orientation of the systems or subsystems, distance, earthing

etc. Reactive coupling comes in the form of either capacitive or inductive coupling. The fact that high voltages (or high impedances) give rise to a large electric (E) field accounts for the capacitive coupling between wires, whereas high currents (or low impedances) will give rise to large magnetic (H) fields which account for inductive coupling. This latter statement can be seen from Faraday's law:

$$\xi = -N\frac{d\phi_B}{dt} \tag{1.3}$$

where it can be seen that the EMF (ξ) induced in a coil is proportional to the negative of the time rate of change of the magnetic flux cutting the coil (ϕ_B). This changing magnetic flux would generally be set up by current fluctuations in a nearby circuit/system.

3. **Radiation**- Most electronic devices will have some part of their circuit forming a loop or a dipole, and it is configurations such as this that will cause the radiation of electromagnetic energy. As was mentioned, conductive emissions tend to be substantial below 30 MHz. Above this frequency these emissions become less noticeable in relation to the radiated emissions which become dominant.

   The problem of compromising radiation is not improving with technology but is actually getting worse. As more and more manufacturers begin to use wireless technology and the frequency of processors in personal computers (PC's) increase, the level of radiation emanating from a system will also increase. If some of the information signal being processed by the system happens to get coupled onto a loop or dipole type arrangement in the circuit, there is a good chance that it will modulate the radiated signal. This can sometimes be demodulated with something as simple as an average AM radio receiver [13]. This problem will be studied in more detail in chapter 3.

These are all methods which can be fatal to a systems security. It is necessary when designing secure systems that each of these mechanisms be reduced as much as possible.

Another mechanism by which information can be obtained is through a technique known as fault induction. If one can induce faults into a system, then these faults may leak information. For example, a system might only send ciphertext depending on whether or not a certain bit in a register is set or clear. If the bit is set, then the system might encrypt the data, however if it is clear then it might send the plaintext in an ordinary manner. If an attacker could find a way of changing the value of this bit, then the system could be easily compromised. This is a fairly simple example but it demonstrates the idea.

Having discussed some of the physical mechanisms through which information can leak from a system, it is clear that the traditional model of figure 1.3 needs to be reviewed. Figure 1.4 shows an updated version which incorporates the above side channels. The emission "other" might include sound or light (e.g. diodes flashing on and off) but also incorporates emissions that have yet to be identified. In fact, there appears to be another form which has not yet been disclosed by the US government. This can be seen in [14] under the section 'Introduction to TEMPEST' where a list of compromising emanations is given with one item redacted and replaced by x's. This may be something to bear in mind for designers of cryptosystems. It is also stated in the same document that the majority of compromising emanations are generated in electromagnetic field form which is probably the reason the US Government has directed considerable resources into the classified TEMPEST program.

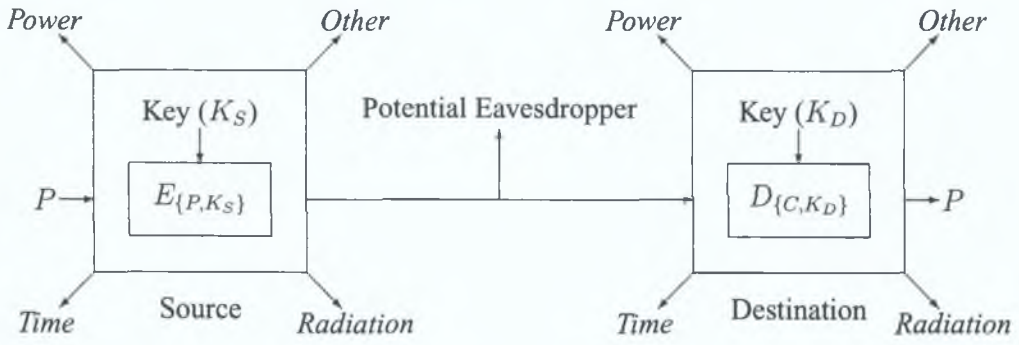


Figure 1.4: Updated model of a cryptographic communication system incorporating side channel emissions. K_S may or may not equal K_D depending on the cryptographic scheme used.

1.2.2 Tamper Resistance

Tamper resistance is the name given to any mechanism which aims to prevent secret information within a physical device from being compromised whilst inside that device. Tamper resistance has been in use for centuries [15]. Techniques such as weighted naval code books (that would sink if required), codes printed in water-soluble ink, and Russian one-time pads printed on cellulose nitrate (this has an explosive flammability and would destroy the pads within seconds if lit) have all been used. Even with these methods in place, a lot of systems have been compromised simply due to the fact that the operators were not vigilant enough. As a result, most modern tamper resistance techniques rely on technology and not on the operator. For example, tamper sensing meshes are placed within the casing of cryptographic devices to detect if an intruder is attempting to drill a hole and enter the system. Building a perfectly tamper resistant device seems like an elusive goal but it is an active area of research at present [16–18].

1.3 FIPS Standards

The main motivation for this thesis is a standard taken from a set of US government standards known as FIPS. The FIPS must be complied with by all US federal government agencies and government contractors processing sensitive but non-classified information. As mentioned already, FIPS is an anagram for Federal Information Processing Standards. The specific standard of concern for this thesis is FIPS 140-2 - "Security Requirements for Cryptographic Modules". The FIPS 140-2 outlines the security requirements that all federal government owned cryptographic modules, processing sensitive data, should comply with. It is a revision of the FIPS 140-1 and supersedes it as of May 2002. However, the FIPS 140-1 still remains valid but no devices have been tested against it since this date.

It might prove useful to have a brief look at the revisions FIPS 140-2 makes reference to [19]:

- Allow separation of plain text from other types of input or output through physically and logically separate ports.
- Strengthen authentication mechanisms and address minimum probabilities for guessing, false acceptance error rates and restrictions on feedback to users.
- Replace the old trusted Computer System Evaluation Criteria with Common Criteria for Information Technology Security Evaluation
- Add requirements for over-the-air re-keying for radio cryptographic modules
- Require four statistical random-number generator tests during self-testing instead of one, with higher statistical limits for random-number generation
- Cover new types of cryptographic attacks that use power or timing analysis or fault induction.

It is the last item on this list that is of concern here. These "new" attacks also cover the area of TEMPEST as mentioned, and all will be discussed in different levels of detail throughout this thesis.

The FIPS 140 set of standards defines 4 different security levels, each more stringent than the preceding one. Level 1 is the lowest and level 4 the highest. Generally the requirements of each level are supersets of each of the levels below it. It is possible for a cryptographic module to meet different levels for different operations, however the overall certified level is taken as the lowest one the module complies with.

The program that validates modules to these cryptographic standards is the Cryptographic Module Validation Program (CMVP) and the laboratories that carry out this validation have to be National Voluntary Laboratory Accreditation Program (NVLAP) accredited. A list of accredited laboratories can be found at [20].

The specific section of this standard under review in this thesis is section 4.11 and is entitled "Mitigation of other attacks". This is a completely new section and provides information about several new types of attacks (each of which has already been discussed). It was developed as a direct result of numerous public comments which suggested that they should be included in the revised standard (i.e. FIPS 140-2). The attacks were included following these comments, however no test procedures were given and in fact are not tested against in the general case. If the vendor of a cryptographic module states in it's security policy that mitigation techniques have been implemented, then the NVLAP laboratory has to test that claim [21]:

> "An additional area concerned with the mitigation of other attacks is currently not tested but the vendor is required to document implemented controls (e.g., differential power analysis, and TEMPEST)."

This research aims to determine the vulnerability of the DUT to these attacks so that

pending tests (of which there are guaranteed to be a few) can be approached with a certain degree of confidence.

1.4 Motivation and thesis aims

In the past decade research on side channel attacks has increased dramatically. Most of the results provided have turned out to be worrying for designers of systems such as smart cards. These are low power, minimal circuitry devices containing a microprocessor and some memory for storage of items such as cryptographic keys and personal information. Smartcards may not be the only devices at risk and it must be considered also that more complex cryptographic systems may also be vulnerable.

The aim of this thesis is to test the vulnerability of a recently developed high speed embedded cryptographic processor to certain side channel attacks. A number of side channel attacks exist but only two were chosen as they appeared to offer the most threat to the current system. With the knowledge gained from these tests, the designers will have an increased understanding of the devices immunity level when attempting to comply with modern cryptographic standards such as the FIPS. Along with an indication of the vulnerability of the device, current countermeasures will also be briefly discussed. The experimental techniques developed can be used as templates for designers wishing to test the vulnerability of their own designs to these attacks.

1.5 Thesis Organisation

This chapter has introduced the concepts that will appear throughout the thesis, and has given an overview of cryptography and compromising emissions. Chapter 2 discusses in detail the area of power analysis. This is vital for an understanding of the experimental results given later. Following this, chapter 3 will discuss Electromagnetic Analysis (EMA) which can also be used to compromise the security of a cryptographic system.

It will be seen in this chapter that some of the attack methods are very similar to that of chapter 2. Chapter 4 discusses the experimental setup, problems encountered and results taken. It will be seen here that the attacks were not completely successful against the DUT but did work against a software simulation and a PIC microcontroller implementation. Chapter 5 briefly discusses some proposed countermeasures for mitigating compromising emanations. Finally, conclusions are drawn on the board's security and comments given on whether or not extra action needs to be taken.

Chapter 2

Power Analysis

Power Analysis is a relatively new concept but has proven to be quite effective in attacking smartcards and similar devices[1]. It was first demonstrated by Ernst Bovelander in 1997 but a specific attack strategy was not given [22]. A year later it was brought to the general public's attention by Paul Kocher and the Cryptographic Research team in San Francisco [23]. Kocher et al. provided an attack strategy that would recover the secret key from cryptographic systems running the DES algorithm. This caused great concern amongst the smartcard community and a search for an effective countermeasure began. To date a limited number of countermeasures have been proposed and none are fully effective. The attacks work equally well on other cryptographic algorithms as shown by Thomas Messerges et al. who presented a great deal of supplementary research on the subject [24–27].

Power analysis involves an analysis of the pattern of power consumed by a cryptographic module as it performs its operations. The purpose of this pattern analysis is to acquire knowledge about causal operations that is not readily available through other sources. The power consumption will generally be different for each operation performed (and even for the same operations with different data values). One of the causes of these variations is the transistor technology used to implement the module. The transistors act as voltage controlled switches, and the power they consume varies

[1] The smartcard is very susceptible to this form of attack mainly because it applies little or no power filtering due to its small size.

with the type of instructions being processed. For example, a conditional branch instruction appears to cause a lot of noticeable fluctuations according to [10], and should therefore be avoided if possible where secret keys are concerned.

This chapter outlines the theoretical aspects of a power attack (details of a practical implementation will be reserved for chapter 4). Although the ideas are the same, the attacks against secret key algorithms and public key algorithms are slightly different. The first section will look at power analysis on symmetric ciphers and an explanation will be given with reference to DES. Following this, the idea will be expanded to include asymmetric ciphers. Specifically, it will look at ciphers that use modular exponentiation (the mathematical operation running on the DUT). Such ciphers include RSA, Diffie-Hellman key agreement, and ElGamal. An understanding of each cipher is not necessary as long as the idea of modular exponentiation and its implementation is understood. The appendices contain material that can be used to assist in this understanding and will be referred to in the appropriate sections.

Power analysis generally refers to attacks carried out using physical connections to the cryptographic modules in order to measure the power fluctuations. However, alternate versions of this attack using electromagnetic radiation can be found in [11, 28]. It is claimed in [28] that these alternate attacks are as good as, if not better, than ordinary power attacks (such as those described in this chapter). This is especially so, if a guess at the key is mistaken. Chapter 3 discusses this in a lot more detail.

Before taking a look at the specific power attacks, it will be instructive to examine the cause of these power fluctuations in a bit more detail. A brief outline is presented here but the reader is referred to [29–31] for more information and a more mathematical treatment of the subject.

2.1 Theoretical foundations of power fluctuations in digital circuitry

The main technology used today to implement digital circuitry, is known as Complementary Metal Oxide Semiconductor (CMOS) technology. CMOS devices are constructed using two complementary Metal Oxide Semiconductor Field Effect Transistors (MOSFETs) - an N-channel MOSFET (NMOS) and a P-channel MOSFET (PMOS). Both transistors can be used separately in digital circuitry fabrication and offer higher packing densities (more transistors per chip) than CMOS. However, CMOS has two main advantages. It offers higher speeds than both PMOS and NMOS and has a very low static power dissipation (lower than the other MOS families) which makes it very attractive in the continuing efforts of power conservation. Unfortunately however, the power dissipation is really only negligible under static conditions (i.e when the input voltage is unchanging). Once transitions occur from one logic level to another, transient currents exist. These transient currents have been proven to be enough to assist adversaries in gaining knowledge about the secret parameters residing on cryptosystems.

Figure 2.1 shows a simple CMOS inverter. It is constructed with a PMOS transistor (upper) and an NMOS transistor (lower). Each transistor consists of a Gate (G), Drain (D) and Source (S). The gates of the two devices are connected together and are therefore a common input. The drains of the two are also tied together and are a common output. The source of the PMOS device is connected to the positive supply voltage (V_{DD}) whereas the source of the NMOS device is connected to ground (V_{SS}).

When V_{in} is HIGH ($V_{in} = V_{DD}$) the PMOS transistor will be switched off and the NMOS transistor will be switched on. The output voltage V_{out} is then LOW (i.e $V_{out} = 0V$) when V_{in} is HIGH (hence the title "inverter"). In this static condition, very little current flows. The undesirable current that does flow is known as *leakage current* and

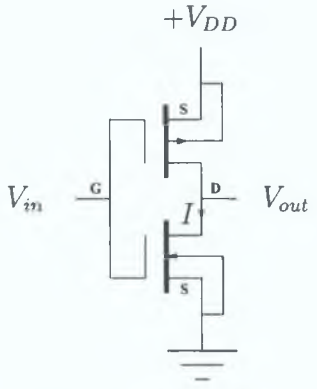


Figure 2.1: CMOS Logic Inverter.

constitutes a small Gaussian noise source (approximately) as far as power analysis is concerned [32]. When V_{in} is LOW, each of these values is reversed and again only the minute "leakage current" flows.

The problem arises when the gates are switched from one logic level to another. As an example, assume that V_{in} is switching from LOW to HIGH. Assume also that the transition is perfectly linear. This will produce the trapezoidal waveform shown in the upper graph of figure 2.2 (adapted from [29]) which has a finite risetime (τ_r) and falltime (τ_f). Once V_{in} reaches a certain threshold voltage (V_{Tn}), the NMOS transistor which was initially off, will begin to turn on. As this happens it's very high equivalent resistance will gradually drop to much lower levels. This will allow a transient current to flow[2] which will increase as V_{in} rises towards half of another threshold voltage $V_{DD} + V_{Tp}$, where $V_{Tp} = -V_{Tn}$ in perfectly symmetric cases. Once the voltage gets beyond this halfway point, the current will begin to drop off again due to the increasing resistance of the PMOS transistor. At the threshold value of $V_{DD} + V_{Tp}$, the current will be back to it's static conditions. This transient current occurs every time the inverter is

[2]The PMOS transistor will already be on at this stage so it won't be causing much resistance to the current. It's equivalent resistance however will begin to increase.

toggled (i.e on both rise and fall sections of V_{in}). The current pulses on the rising and falling edges of V_{in} both have exactly the same height and shape as can be seen.

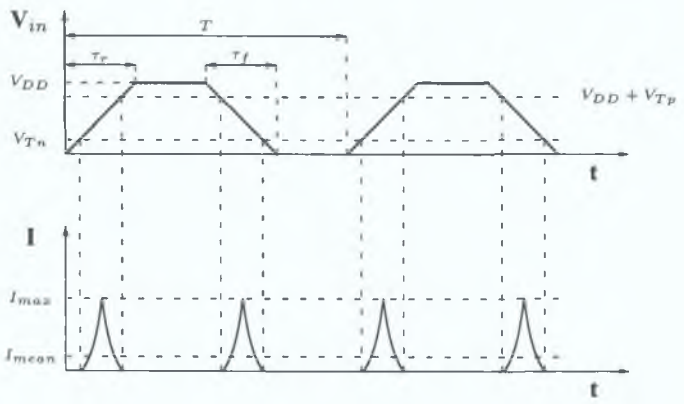


Figure 2.2: Current behaviour of inverter without load.

Figure 2.3 shows a PSpice simulation of the circuit in figure 2.1. The upper graph shows the input voltage V_{in} whereas the lower graph shows the current I entering the NMOS transistor. It can be seen again that I is exactly the same on both the rising and falling edges (as is to be expected from figure 2.2) and would therefore be of no use to an attacker - this however is an ideal situation.

In reality, a CMOS device almost always has some sort of capacitive load connected at it's output and the ideal situation of figure 2.2 and 2.3 does not exist in reality. Examples of this capacitive load might be the input to some other digital logic or to some sort of data bus (generally it is an unintentional capacitance however). It might not have been considered before that a data bus is a form of capacitance but in fact it is and very much so. As well as it's own capacitance, the bus generally has other devices attached to it. The bus capacitance thus represents the cumulative effect of all the parasitic capacitances of the different inputs and outputs that are connected it. For simplification, the total capacitance the CMOS logic experiences at it's output can be

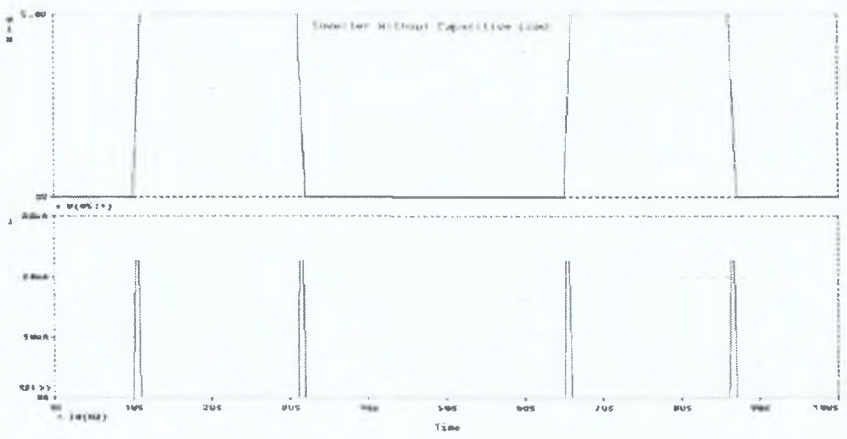


Figure 2.3: PSpice analysis of CMOS logic inverter without load.

represented as a load capacitance C_L. Figure 2.1 can therefore be modified to include this capacitance as shown in figure 2.4.

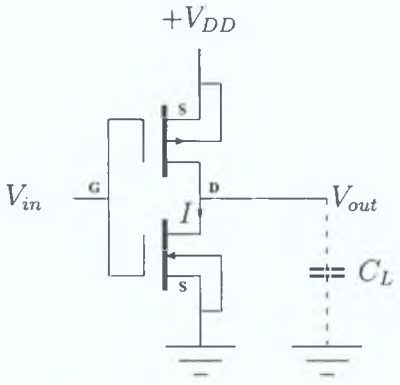


Figure 2.4: CMOS Logic Inverter with capacitive load.

From basic circuit theory, it is known that if the voltage across a capacitor changes, then the current must flow to charge or discharge the capacitor. This is represented by equation 2.1.

$$I_{C_L} = C_L \frac{dV_{out}}{dt}. \tag{2.1}$$

The current I in figures 2.1 and 2.4 will therefore form only part of the total current I_{tot} that is drawn from the supply:

$$I_{tot} = I + I_{C_L}. \tag{2.2}$$

This leads to two separate forms of power dissipation - *Dynamic Dissipation* and *Short-circuit dissipation*.

Dynamic power dissipation is due to the charging and discharging of the capacitive load. It is given by the equation

$$P_D = V^2 f C_L. \tag{2.3}$$

It is the more dominant of the two forms and as can be seen, is directly proportional to the switching frequency. It is independent of the parameters of the transistors.

Short circuit dissipation is due to a short circuit path being created when both the PMOS and NMOS transistors are switched on. It can be represented by

$$P_s = V I_{mean} \tag{2.4}$$

where I_{mean} depends on the switching frequency and parameters of the transistor such as the rise and fall time.

Attackers can use these currents to attack systems as follows. Assume a LOW is initially applied to the input of the logic gate ($V_{in} = V_{SS} = 0V$) of figure 2.4 for a long period of time. The output voltage V_{out} will therefore be at the supply poten-

tial and the capacitor will be fully charged. Once a HIGH is applied to the input (i.e $V_{in} = V_{DD} = 5V$), the output will begin to drop to a LOW and the transient current produced would cause the capacitance to discharge through the NMOS capacitor. As a result, more short circuit current would flow than the case where the logic gate had no load attached (figure 2.1). If the input to the logic gate is now switched back to a LOW value, the output will jump from a LOW value to a HIGH value and the capacitor would charge up again. This would cause less short circuit current to flow than for the previous case (due to dynamic power dissipation). The difference in this current level could be used to determine the different logic levels being used and can thus reveal secret parameters stored on the DUT.

Figures 2.5 and 2.6 show two more PSpice simulations but this time with a capacitive load attached to the output of the inverter. The first capacitive load is .1pF whereas the second one is 10 times this or 1pF. It can be seen that the larger the load capacitance the more the current transients differ due to the larger charge required to keep the output voltage at the same levels. It is prudent therefore to try to keep the load capacitance to a minimum.

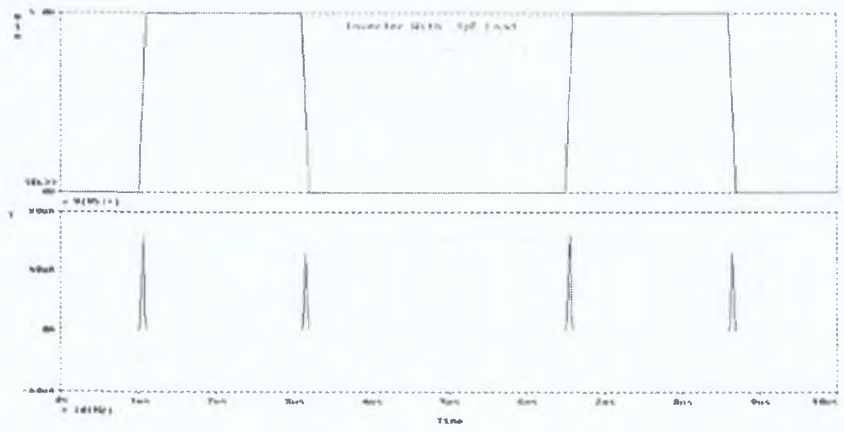


Figure 2.5: PSpice analysis of CMOS logic inverter with .1pF load.

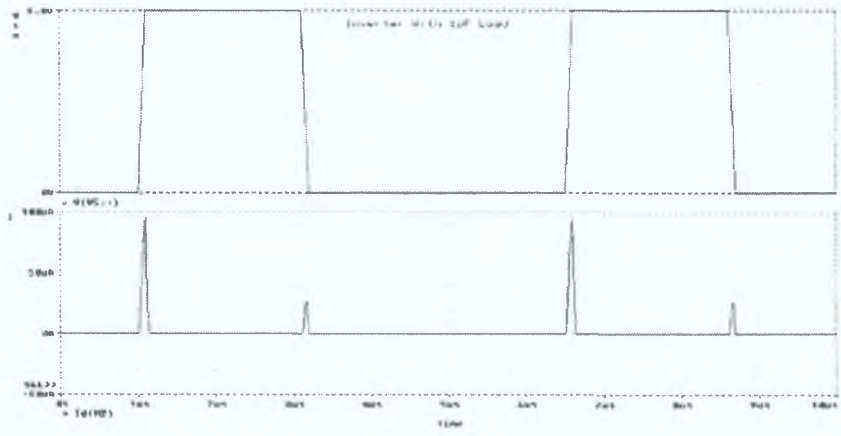


Figure 2.6: PSpice analysis of CMOS logic inverter with 1pF load.

The next section describes in detail how the difference in these transient currents can be used to attack physical implementations of cryptosystems and to learn about the operations they are running.

2.2 Overview of Power Attacks

Power analysis attacks can be divided up as follows:

1. Simple Power Analysis (SPA)
2. Differential Power Analysis (DPA)

    Single bit DPA

    Multiple bit DPA

    High Order DPA

    Inferential Power Analysis (IPA)

SPA attacks are the easiest to implement and take the order of seconds to minutes (once certain characterisation tests have been carried out [24]) to perform however,

a cryptosystem will generally not be directly compromised by these attacks. DPA attacks on the other hand, can take hours to days[3] to perform but will generally reveal a great deal more information than an SPA attack and can ultimately compromise a cryptosystem. Both attacks will generally be used together as will be seen in chapter 4.

2.3 Simple Power Analysis (SPA)

Simple Power Analysis involves measuring the power fluctuations of a cryptosystem using some form of digitising oscilloscope, and using visual inspection to try to determine information about that system. There would be little or no formal mathematical analysis performed on this data however. The amount of information attainable depends in part upon the implementation of the algorithm being attacked.

For smartcards and similar devices, the power can be measured across a $10 - 50\Omega$ resistor[4] in series with the power or ground line of the specific device. However, it is not so simple on a complex cryptographic processor which probably draws its power from the peripheral component interconnect (PCI) backplane of a computer.

Ideally, the attacker would wish to get as close as possible to the actual chip performing the operations if a high signal to noise ration (SNR) is to be obtained. This might be more difficult than it first appears as information on which of the boards numerous chips is actually running the algorithm may not be readily available. Even if it were, the power pin of the chip would have to be physically separated from the board to perform the attack and then reattached once complete (if the attack were to go unnoticed). Most tamper resistance devices would not permit this from happening. The method used in

[3]The length of time will be determined by (amongst other things) the speed of the particular DUT, the size of the cryptographic key, and the measurement instrumentation used to capture the waveforms. It could take weeks if extremely large quantities of data are to be processed.

[4]The resistor should be small enough so as not to interfere with the operation of the circuit itself, but large enough to give easily observable voltage fluctuations.

this research is given in chapter 4, and it will be seen that it allows a non-destructive way of getting quite close to the area of interest (even if it is not at the particular chip itself).

If DES is poorly implemented it can be quite vulnerable to SPA attacks [24]. These can be used to view operations such as individual DES rounds and even actual microprocessor instructions [10, 33]. For example, it might be possible to tell whether or not a jump instruction has been performed[5]. This information can then be used to determine the secret information within a system.

The main information that can be retrieved using SPA is the overall structure of the cryptographic code. It would be very easy to tell how long it takes the cryptoprocessor to complete one full encryption. It will also be quite easy to see events that are repeated such as the rounds of DES or each exponent bit operation in a public key system. Other timing values can be gained from these which can be used to improve the efficiency of the more complex DPA attack.

Two other forms of possible information leakage using SPA are *Hamming Weight Leakage* and *Transition Count Leakage* [24] and they are discussed next.

2.3.1 Hamming Weight Leakage

The *Hamming Weight* of any code vector is defined as the number of nonzero elements in the vector. This is a general definition where the alphabet[6] may contain more than two values. For binary, this simply means the number of 1's in a word of data.

If the Hamming Weight of the secret key is known, a brute force attack might be a lot easier as the key space needed to be searched is reduced in size. In [26], Messerges et al. prove that the key search space is reduced from 2^{nk} down to

[5]This may be true for low speed, minimal circuitry devices such as smartcards but its likelihood decreases with more complex, high speed cryptoprocessors.

[6]The symbols used in a particular system such as 1 and 0 in a binary system.

$$\left(\sum_{m=0}^{n}\binom{n}{m}^2/2^n\right)^k \tag{2.5}$$

where n is the number of bits in a word of data, k is the number of words, and m is the Hamming weight value.

For example, consider the byte of data 10110100_2 and assume that it is a secret key. The Hamming weight of this key is 4 as there are 4 nonzero elements. Generally, it would take an attacker $2^8 = 256$ tries[7] before the secret key is found. However, if the Hamming weight is known, then by equation 2.5, it can be seen that this value is reduced down to 50.27 keys (or an equivalent key size of 5.65 bits). A more practical example of it's threat is given in [26] where it is shown that the key search space for the DES algorithm is reduced from 2^{56} down to about 2^{40}; this is quite a significant reduction.

Knowing the Hamming weight of a key really offers no significant advantages over much larger key sizes (such as the $2,176$ bits used in the DUT here). In this case, the key search space with known Hamming weight, would still be far to large to be searched by any computing power available today, in any reasonable amount of time. However, Messerges et al. claim that if the key is rotated once each round (as it is in DES see appendix A), then enough information can be obtained to completely uncover the secret key [24]. It is claimed then, that even larger key sizes may be vulnerable to attack in this case. As will be seen though, asymmetric algorithms would not be vulnerable to this attack as there is no key rotation involved.

2.3.2 Transition Count Leakage

Transition Count is the Hamming weight of the difference (also known as the *Hamming distance*) between the Hamming weight of the data on the bus at one point and that of

[7] Again, on average, only half this value would need to be searched.

the data following it.

When the information on the data bus changes state, a large number of logic gates will conduct current. The greater the Hamming distance between the word of data being placed on the bus, and the one already there, the greater the power dissipated. This power can be recorded and measured as usual. It's usefulness in an attack stems from the fact that at certain times, the contents of the bus might be known to the attacker (it might be some regularly accessed address or opcode). If this is the case, the attacker would then be able to use a power trace obtained with known bus contents, and compare them to another trace obtained with unknown contents (such as parts of the secret key). The difference would give an idea as to what the unknown contents might be. This could be improved by comparing a number of traces where the contents are known to the one trace where the contents are unknown. The combined results could then be used to reveal the entire subkey under attack.

Transition count leakage has a more limited application than Hamming Weight leakage. It may be more difficult to implement, due to the fact that the attacker would need to know the contents of the data bus, either before or after the attack. If the source code was not readily available, then the attacker might have to micro-probe the device in order to obtain this information. Again, tamper resistance devices may not permit this from happening.

From the explanation of power analysis and transient switching, it would seem that only transition count leakage might be observed by an attacker. According to [34] however, this is not the case due to the fact that the bus will conduct current when a word of data is placed on it. This current can be just as clearly visible to an attacker as the current increase produced by switching transistors.

2.3.3 The Threat of SPA

Kocher et al. claim that SPA attacks are of little menace to the smartcard community and that they are easy enough to prevent. However, in [34], Rita Mayer-Sommer shows that this is not really the case. It is claimed that although SPA may not be as threatening as DPA (in the sense that less information is revealed), the fact that DPA is a lot more experimentally intensive means that relatively speaking, SPA is the more threating attack. A great deal of information can be gained from one power trace, whereas in DPA, hundreds, maybe even thousands of power traces may need to be collected.

The experiments in [34] were carried out on a low power, low frequency, PIC[8] microcontroller. The specific chip used was the PIC 16C84 which was run at 4MHz and 4.5V. One of the claims made in this paper was that it is not necessary to average noise out of the consumption characteristics provided the device is run at a low enough frequency and high enough supply voltage. This would seem to suggest that SPA attacks on complex high speed cryptographic devices might be much more difficult.

Although SPA attacks may not reveal the cryptographic key in more complex systems (including systems with larger key sizes), it should not be dismissed as being a harmless attack. There is a great deal of knowledge that an attacker can gain using SPA alone such as the timing information discussed above. Again, actual results are presented in chapter 4.

2.4 Differential Power Analysis (DPA)

Differential power analysis is a much more powerful attack than simple power analysis, and can reveal a lot more information. The extra price for this information comes from the fact that it's implementation requires a great deal more knowledge and skill. This knowledge can range from transistor design up to the specific cryptographic algo-

[8] Apparently this is not an anagram.

rithms used, and the attacker needs to be quite competent in such areas as electronics, statistics, cryptography and programming.

If SPA was used on a smartcard device, it would only be possible to see the combined power consumption of every circuit element on the card. However, fluctuations caused by individual switching transistors would be impossible to see. Using DPA, the statistical methods utilised make it possible to observe these minute fluctuations - although it becomes increasingly difficult as the system becomes more complex (this is especially true for systems with highly parallel architectures).

A DPA attack is relatively cheap to implement and can be automated once an understanding of a device has been achieved. According to Kocher [10], one of the main reasons systems are vulnerable to these form of attacks is because designers of the different system levels do not interact with each other. As a result, assumptions are made about the security effectiveness of the rest of the design. This problem would certainly need to be addressed by companies developing cryptosystems if any progress is to be made in mitigating these attacks.

To implement DPA, quite a few encryption transactions need to be executed, and the resulting power traces recorded. Certain bits of the key will be guessed, and correlation techniques in the form of a *differential trace* will be used to determine whether or not this guess is correct. If it is, small peaks will be seen in the differential trace at each instant a target bit was manipulated. However, if it is not, the trace will tend to zero or some steady value[9].

The traces must contain as many sample points as is practical. The more samples obtained within a fixed time period, the higher the probability of success (to a limit of course). This will lead to greater processing times however. The specifics of the

[9]This may not entirely be true as there may be statistical biases within the algorithm itself that will cause small peaks in traces with incorrect guesses. However these peaks will be a lot smaller than those within the trace of a correct guess, and should not cause a problem for the attacker.

acquisition system (memory, speed etc.) will determine both of these parameters and a compromise must be made (i.e. more sample points and slower processing times or vice versa?).

Details of the attack will now be outlined with reference to the symmetric algorithm DES, and then expanded to include asymmetric algorithms in the section that follows. It will be seen that there are many similarities between the two, but also a number of distinct differences. Although the following explanation is with reference to a specific symmetric algorithm, it can easily be modified to attack other similar types of algorithms.

2.4.1 DPA attacks on a symmetric key algorithm - DES

A DPA attack on DES can be carried out with either known plaintext or known ciphertext. If the plaintext is known, the attack would concentrate on the DES's first round[10] but if the ciphertext is known, then it's last round would be attacked. The reasons for this will become clear once this section is understood. For the following explanation, a known ciphertext attack will be examined but the same methods are also applicable to a known plaintext attack.

As shown in the list at the beginning of the chapter, DPA can be split into single bit DPA, multiple bit DPA, High-order DPA and IPA. DPA will initially be explained with reference to single bit DPA. This is then expanded for improved results using multiple bit DPA and high-order DPA. IPA is an attack developed independently of the other three and appears to have advantages over normal DPA however, it's relative success level is unclear at present. It is included here for completeness.

[10]See appendix A of details of DES and it's "rounds".

2.4.1.1 Single Bit DPA

In order to carry out a DPA attack, a method must be devised to produce a random set of J plaintext inputs that can be sent to the cryptosystem for encryption[11]. On receiving these plaintext inputs, pi_j, $1 \leq j \leq J$, the board will begin to run its algorithm and draw varying amounts of power. These power fluctuations can be sampled using a digital sampling oscilloscope which should be capable of sampling at about 20-30 times the clock frequency being used[12]. This can amount to extremely high sampling rates for high speed cryptographic processors as will be seen in chapter 4.

The waveforms observed for each pi_j can be represented as a matrix wf_{jk}[13], where $1 \leq k \leq K$. A second column matrix, co_j, can also be used to represent the ciphertext output. In practice, each row of wf_{jk} would probably be stored as a separate file for ease of processing. Considering the plaintext inputs as a column matrix also, the three can be brought together in tabular format for reference:

Input	Sampled Waveform					Output
pi_1	wf_{11}	wf_{12}	$\cdots$	$wf_{1(K-1)}$	wf_{1K}	co_1
pi_2	wf_{21}	wf_{22}	$\cdots$	$wf_{2(K-1)}$	wf_{2K}	co_2
pi_3	wf_{31}	wf_{32}	$\cdots$	$wf_{3(K-1)}$	wf_{3K}	co_3
$\vdots$			$\vdots$			$\vdots$
$pi_{(J-2)}$	$wf_{(J-2)1}$	$wf_{(J-2)2}$	$\cdots$	$wf_{(J-2)(K-1)}$	$wf_{(J-2)K}$	$co_{(J-2)}$
$pi_{(J-1)}$	$wf_{(J-1)1}$	$wf_{(J-1)2}$	$\cdots$	$wf_{(J-1)(K-1)}$	$wf_{(J-1)K}$	$co_{(J-1)}$
pi_J	wf_{J1}	wf_{J2}	$\cdots$	$wf_{J(K-1)}$	wf_{JK}	co_J

Table 2.1: The cryptosystem accepts a plaintext input (pi_j) and produces both a power waveform (wf_{jk}) and a ciphertext output (co_j). Sometimes only the ciphertext is available to the attacker.

Having captured each power waveform and ciphertext output, a function known as a *partitioning function*, D(.), must now be defined. This function will allow division

[11] This method must be automated as the number of random plaintext inputs will be quite large.

[12] This is not a theoretical limit as in Shannon's sampling theorem but it is recommended for a successful attack.

[13] The subscripts j and k are used to identify the plaintext number causing the waveform and the time sample point within that particular waveform, respectively.

of the matrix wf_{jk} into two sub-matrices $wf0_{pk}$ and $wf1_{qk}$ containing P and Q rows respectively, with $1 \leq p \leq P$ and $1 \leq q \leq Q$ where $P + Q = J$. Provided that the inputs pi_j were randomly produced, then $P = Q = J/2$ as $J \rightarrow \infty$ (i.e. the waveforms will be divided equally between the two sets).

No dependable variables are explicitly stated for the partitioning function here as they may vary slightly from algorithm to algorithm (and indeed from one partitioning function to another within the same algorithm). It must however depend on some part of the key[14], as this is the information sought.

The partitioning function allows the division of wf_{jk} because it calculates the value of a particular bit, at particular times, during the operation of the algorithm. If the value of this bit is known, then it will also be known whether or not a power bias should have occurred in the captured waveform. For a 1, a bias should occur, and for a 0 it shouldn't. Separating the waveforms into two separate matrices (one in which the bias occurred and another in which it didn't) will allow averaging to reduce the noise and enhance the bias (if it occurred). For randomly chosen plaintexts, the output of the D(.) function will equal either a 1 or 0 with probability $\frac{1}{2}$ (this is just another statement of the fact that $P = Q = J/2$ as $J \rightarrow \infty$).

Messerges et al. give an example of a partitioning function in [24, 26] as

$$D(C_1, C_6, K_{16}) = C_1 \oplus SBOX1(C_6 \oplus K_{16}) \tag{2.6}$$

where SBOX1(.) is a function that outputs the target bit of S-box 1 in the last round of DES (in this case it's the first bit), C_1 is the one bit of co_j that is exclusive OR'ed with this bit, C_6 is the 6 bits of co_j that is exclusive OR'ed with the last rounds subkey and

[14]No variable representing the key is given within the parenthesis here as this might imply dependence on the entire key. This however will never be the case, as this would then be no different from a brute force attack as will be seen.

K_{16} is the 6 bits of the last round's subkey that is input into S-box 1.

The value of this partitioning function must be calculated at some point throughout the algorithm. So, if the values C_1, C_6 and K_{16} can be determined, it will be known whether or not a power bias occurred in each waveform. The values C_1 and C_6 can be determined from each ciphertext in table 2.1. The value of the subkey K_{16} is the information sought. To find this, an exhaustive search needs to be carried out. As it is 6 bits long, a total of $2^6 = 64$ subkeys will need to be tested. The right one will produce the correct value of the partitioning bit for every plaintext input. However, the incorrect one will only produce the correct result with probability $\frac{1}{2}$. In this case, the two sets $wf0_{pk}$ and $wf1_{qk}$ will contain a randomly[15] distributed collection of waveforms which will average out to the same result. The differential trace (discussed below) will thus show a power bias for the correct key only. Of course it means that 64 differential traces are needed but this is a vast improvement over a brute force search of the entire 56 bit key.

Mathematically, the partitioning of wf_{jk} can be represented as

$$wf0_{pk} = \{wf_{jk} | D(.) = 0\} \tag{2.7}$$

and

$$wf1_{qk} = \{wf_{jk} | D(.) = 1\} \tag{2.8}$$

Once the matrices $wf0_{pk}$ and $wf1_{qk}$ have been set up, the average of each is then taken producing two waveforms $awf0_k$ and $awf1_k$ both consisting of K samples. By taking the averages of each, the noise gets reduced to very small levels but the power spikes in $wf1_{pk}$ will be reinforced. However, averaging will not reduce any periodic noise

[15] Provided the plaintext inputs are randomly chosen.

contained within the power waveforms and inherent to the operations on the cryptographic board. This can largely be eliminated by subtracting $awf0_{pk}$ from $awf1_{qk}$ (this can be thought of as demodulating a modulated signal to reveal the "baseband", where the periodic noise is the "carrier"). The only waveform remaining will be the one with a number of bias points identifying the positions where the target bit was manipulated. This trace is known as a *differential trace*, ΔD_k.

Again, in mathematical terms, the above can be stated as

$$awf0_k = \frac{1}{P} \sum_{wf_{jk} \in wf1} wf_{jk} = \frac{1}{P} \sum_{p=1}^{P} wf0_{pk} \tag{2.9}$$

and

$$awf1_k = \frac{1}{Q} \sum_{wf_{jk} \in wf0} wf_{jk} = \frac{1}{Q} \sum_{q=1}^{Q} wf1_{qk} \tag{2.10}$$

The differential trace ΔD_k is then obtained as

$$\Delta D_k = awf1_k - awf0_k \tag{2.11}$$

The last five equations can now be condensed into one [10],

$$\Delta D_k = \frac{\sum_{k=1}^{K} D(.) wf_{jk}}{\sum_{k=1}^{K} D(.)} - \frac{\sum_{k=1}^{K} (1 - D(.)) wf_{jk}}{\sum_{k=1}^{K} (1 - D(.))} \tag{2.12}$$

As a side point, it is very important that each acquisition of the power trace be started at the exact same point, and sampled at the same rate. If this is not the case, averaging will not work, and will in fact decrease any desirable fluctuations that may be present. Simple Power Analysis can be used initially to identify the structure of the waveforms, and to offer some idea as to where to start sampling. Once this is done, precise trig-

gering and trigger delay[16] options must be used to capture the waveform over the time period of interest. This will be discussed in more detail in chapter 4.

As $J \rightarrow \infty$, the power biases will average out to a value ϵ which will occur at times k_D - each time the target bit D was manipulated. In this limit, the averages $awf0_k$ and $awf1_k$ will tend toward the expectation $E\{wf0_k\}$ and $E\{wf1_k\}$, and equations 2.11 and 2.12 will converge to

$$E\{wf1_k\} - E\{wf0_k\} = \epsilon, \qquad \text{at times} \quad k = k_D \tag{2.13}$$

and

$$E\{wf1_k\} - E\{wf0_k\} = 0, \qquad \text{at times} \quad k \neq k_D \tag{2.14}$$

Therefore, at times $k = k_D$, there will be a power bias ϵ visible in the differential trace. At all other times, the power will be independent of the target bit and the differential trace will tend towards 0.

The above will only work if the subkey guess was correct. For all other guesses the partitioning function will separate the waveforms randomly, and equations 2.13 and 2.14 will condense to

$$E\{wf1_k\} - E\{wf0_k\} = 0, \qquad \forall k \tag{2.15}$$

As mentioned above, 64 differential traces are needed to determine which key is the correct one. Theoretically, the one containing bias spikes will allow determination of the correct key however, in reality the other waveforms will contain small spikes due to factors such as non-random choices of plaintext inputs, statistical biases in the S-boxes

[16]This is the length of time the oscilloscope waits to start sampling after receiving a valid trigger.

and a non-infinite number of waveforms collected. Generally however, the correct key will show the largest bias spikes and can still be determined quite easily.

The other 42 bits from the last round's subkey can be determined by applying the same method to the other 7 S-boxes[17]. A brute force search can then be used to obtain the remaining 8 bits of the 56 bit key.

2.4.1.2 Multiple Bit DPA

The last section looked at single bit DPA in which the output of the D function was one of 4 bits. The idea of attacking more than one of these bits at a time was briefly mentioned by Kocher et al. in [10] however no experimental procedure was given. Goubin et al. [35] discuss the idea in a bit more detail but Messerges et al. give a complete analysis of the procedure and provide some nomenclature - *Multiple bit DPA*. This attack is really just an extension of single bit DPA but increases the height of the bias spikes in the differential trace.

For Multiple bit DPA , instead of guessing what the 1st bit of the output of S-box 1 might be, a guess might be made at the entire 4 bit output[18]. If the answer is correct, then the bias signal will be 4 times greater than that in single bit DPA. So, assuming that the mean value of the bias spike was ϵ, the bias spike for multiple bit DPA will be 4ϵ. This makes sense because when 4 bits change, the cryptoprocessor will draw 4 times more current than when one bit changes. As will be seen however, Multiple bit DPA does not always improve the SNR of the differential signal for the same amount of collected waveforms.

Assuming the same D function is the same as that discussed above, one particular attack of this type is known as an "all or nothing" attack. Here, the waveforms are

[17]The same J power signals can be used for each S-box as the different D functions re-order them accordingly.

[18]It is not necessary to chose 4 bits and any number of bits may be guessed at.

partitioned as follows:

$$wf0_{pk} = \{wf_{jk} | D(.) = 0000\}, \tag{2.16}$$

$$wf1_{qk} = \{wf_{jk} | D(.) = 1111\} \tag{2.17}$$

and

$$wf2_{rk} = \{wf_{jk} | D(.) \neq 1111 \quad \text{or} \quad 0000\}. \tag{2.18}$$

where $1 \leq r \leq R$ and R is the number of rows in $wf2_{rk}$. This is known as an all or nothing attack due to the fact the entire output of the D function is guessed at. If the maximum output of the D function were 8 bits instead of 4 then the waveforms above would be partitioned in accordance with $D(.) = 00000000$ and $D(.) = 11111111$.

The matrix $wf2_{rk}$ is discarded and only $wf0_{pk}$ and $wf1_{qk}$ are used to form the differential average trace as before. This is the downfall of this particular attack as it means not all the power signals will be usable. Although the bias spike is approximately 4 times that of a single bit attack, the attacker may need to collect a greater amount of power signals to provide the same SNR as a single bit attack (see table 2.2 below).

It is not necessary of course to attack all the bits output from the D function - any number of them may be attacked. A more general partitioning scheme is given in [26] where it is also shown that the average fraction of power signals that are usable in this general d-bit DPA attack (d being the number of bits attacked) is

$$\begin{array}{ll} \left[2\sum_{k=d}^{n}\binom{n}{k}\right]/2^n & \text{if} \quad d > \frac{n}{2} \\ 1 & \text{if} \quad 0 < d \leq \frac{n}{2} \end{array} \tag{2.19}$$

where n is the word size of the processor (must be even).

Table 2.2 (also taken from [26]) shows the number of waveforms required to keep the SNR equal to that of a one bit DPA attack. It can be seen that attacking between two and six bits of the output requires less power waveforms, and Multiple bit DPA has an advantage over single bit DPA. However for seven and eight bits, an increased number of waveforms are needed and this advantage is lost. The value N_1 is the number of bits required for a one bit DPA attack.

d:	1	2	3	4	5	6	7	8
N_d:	N_1	$.5N_1$	$.44N_1$	$.5N_1$	$.64N_1$	$.88N_1$	$1.3N_1$	$2N_1$

Table 2.2: Number of waveforms required in a multiple bit DPA attack to keep the SNR the same as a single bit DPA attack. N_1 is the number of waveforms required for a one bit DPA attack.

2.4.1.3 High-Order DPA (HO-DPA)

High-Order Differential Power Analysis (HO-DPA) is slightly more complicated than ordinary Differential Power Analysis and is briefly mentioned by Kocher et al. in [36]. It involves getting signals from multiple sources in a system and/or using different measuring techniques to obtain these signals. It may also involve correlating multiple sub-operations within a single waveform. A system that is vulnerable to HO-DPA is also vulnerable to DPA. It is therefore not of much interest to most attackers as DPA is easier to implement. However, in developing mitigation techniques, both attacks should be considered. An example of a second order DPA attack is given in [25] where Messerges et al. use it to attack a software implementation that is resistant to DPA techniques. For this particular research it will be seen that it was necessary to implement some high-order techniques.

2.4.1.4 Inferential Power Analysis (IPA)

IPA is a more recent form of power attack that was introduced by Paul Fahn and Peter Pearson [37] at the Cryptographic Hardware and Embedded Systems (CHES) conference in 1999. It is unclear how powerful a variant of DPA this attack is, as this is the only paper (to the author's knowledge) that discusses IPA. However, it should be briefly mentioned for completeness. The discussion will be completely intuitive. A more mathematical approach can be obtained from their paper.

The attack involves two stages. The first consists of a profiling stage and the second a key extraction stage. The profiling stage is the most computationally intensive of the two but is only required once. Its purpose is to find where in time the algorithm actually handles the key bits being sought. Once this information is acquired, it can then be used to attack other devices with the same hardware and algorithmic structure using a lookup table constructed from the profiling stage.

The first step involves acquiring hundreds of traces using different plaintext values and a constant key (other variations are possible also and some are given in their paper). The traces are then averaged to reduce noise and data dependent fluctuations; so far this is similar to an ordinary DPA attack. Once the traces are averaged, a number of periodic sections will be noticed that will correspond to each of the "rounds"[19] within the algorithm. Each of these rounds can then be extracted from the main waveform and averaged amongst themselves; this removes fluctuations caused by key values and is known as a *super-average*. The remaining fluctuations are then simply due to specific features of the code. This super-average, can then be subtracted from each of the extracted "rounds" leaving only the key dependent power fluctuations for each round. The resulting traces can then be squared and averaged to produce clean peaks at the

[19] In this case, "rounds" refers to some periodic repetition of a particular section of the code. For the attack to be successful, it is necessary that this periodic repetition exist.

points where key bits were manipulated.

The second step is a lot less complicated and is used for all subsequent attacks. A table is constructed which lists the positions in each round where the fluctuations corresponding to the key bits are located and which particular bit each fluctuation corresponds to. By simply obtaining a single trace the key values can be read using the table.

A particular strength of IPA over DPA for symmetric algorithms is that it is capable of determining the bit's of the secret key from any point in the entire algorithm. In DPA however, only the first or last round can be attacked. This is advantageous to the attacker as some of the methods employed as countermeasures rely on the fact that only the first or last round can be attacked using ordinary DPA.

2.5 Variations of the DPA attack for public key systems

Although DPA attacks on symmetric ciphers have been described, modern cryptography[20] makes use of both symmetric and asymmetric systems for encryption. The DUT was capable of encryption using both types of algorithms however access to the symmetric encryption facility was not provided. As a result the main attacks carried out on the board were against the asymmetric methods employed. This section describes these attacks and supplements the knowledge obtained in the last section. The main body of work in this area is credited to Messerges et al. and is contained in [27].

2.5.1 Implementation of modular exponentiation

As mentioned in Chapter 1, most public key schemes use the process of modular exponentiation in one form or another. This is a computationally intensive operation and requires special techniques for it's implementation. Two methods are generally used

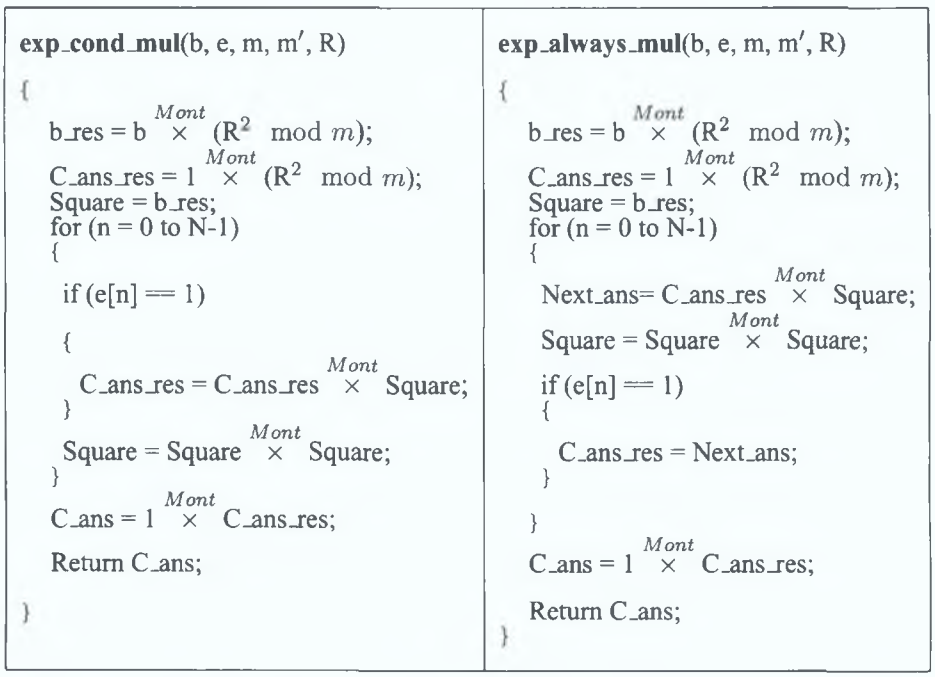
[20] As described in chapter 1 this is from about 1976 on.

in conjunction with each other. The first carries out the actual exponentiation and is known as the *square and multiply* algorithm or *binary method*. It was proposed by the inventors of RSA as an efficient way to implement the exponentiation section of their algorithm. It is studied in a lot more detail by Knuth [38]. The second method was developed in 1983 by Peter L. Montgomery and is known as *Montgomery reduction* [39]. It provides an efficient method for the modular multiplication of two numbers by allowing the modular reduction to be carried out without division by the modulus m. Utilising Montgomery reduction within the binary method allows modular exponentiation to be carried out very efficiently. A detailed explanation of Montgomery reduction (and the notation used to represent it) is described in appendix D.

The binary method calculates the value of b^e, where b is the base and e is the exponent. There are two main versions of the algorithm which operate on the exponent in different ways. The first works from its most significant bit (MSB) down to its least significant bit (LSB) and is referred to as the left to right, square and multiply algorithm. However, the second operates from its LSB to its MSB and is thus known as the right to left, square and multiply algorithm. Due to the requirement of storing an additional intermediate result, the second version is generally the least preferred of the two [40]. Despite this fact, it was this version used on the DUT.

Figure 2.7 shows two versions of the right to left square and multiply algorithm using Montgomery's method. The function **exp_cond_mul()** only performs the multiplication (i.e. C_ans_res $\overset{Mont}{\times}$ Square) if a particular condition is satisfied. However, the function **exp_always_mul()** always performs it, regardless of this condition. The **exp_cond_mul()** is the more efficient method but, as will be seen later, **exp_always_mul()** is the best method for mitigating power attacks. The function **exp_always_mul()** is similar to the method utilised by the DUT.

The inputs and output are the same for both versions of the algorithm as would be



exp_cond_mul(b, e, m, m′, R)

```
{
  b_res = b  ×^Mont  (R^2 mod m);
  C_ans_res = 1  ×^Mont  (R^2 mod m);
  Square = b_res;
  for (n = 0 to N-1)
  {
   if (e[n] == 1)
   {
    C_ans_res = C_ans_res  ×^Mont  Square;
   }
   Square = Square  ×^Mont  Square;
  }
  C_ans = 1  ×^Mont  C_ans_res;
  Return C_ans;
}
```

exp_always_mul(b, e, m, m′, R)

```
{
  b_res = b  ×^Mont  (R^2 mod m);
  C_ans_res = 1  ×^Mont  (R^2 mod m);
  Square = b_res;
  for (n = 0 to N-1)
  {
   Next_ans= C_ans_res  ×^Mont  Square;
   Square = Square  ×^Mont  Square;
   if (e[n] == 1)
   {
    C_ans_res = Next_ans;
   }
  }
  C_ans = 1  ×^Mont  C_ans_res;
  Return C_ans;
}
```

Figure 2.7: Two possible versions of the "right to left, square and multiply" algorithm. The function **exp_always_mul()** is more like the one implemented by the DUT and is more resistant to DPA attacks. Both versions use Montgomery's method for modular reduction to increase efficiency.

expected. The main inputs are the base b, the exponent e and the modulus m. The other inputs (m' and R) are used for the Montgomery operations and are described in appendix D. The output is the value of $b^e \bmod m$. As can be seen, the algorithm operates on each bit separately and enters a conditional statement depending on whether or not the current bit is a 1 or a 0. If the conditional statement is satisfied, **exp_cond_mul()** will perform a Montgomery multiplication as well as a register update however, **exp_always_mul()** will only perform a register update.

2.5.2 Explanation of compromising power fluctuations

The implemented version of the algorithm **exp_always_mul()** was slightly different from that given in figure 2.7. The main point to note is that it was implemented in hardware and not software. This allowed for a highly parallel approach which intro-

duced a great deal more synchronous noise[21] into the system and increased the level of difficulty of the attacks. The multiply operation was performed on every round regardless of the value of the exponent. This occurred in the same clock cycle as the square operation. The general attack on the square and multiply algorithm relies to an extent on the multiply only being performed in accordance with the value of each bit in the exponent. This is because a great deal more power is required for both the multiply and the register update, than that for only the register update. This will be clearly visible in a differential trace. Due to the fact the DUT didn't implement it this way, it was expected that the attack would be a great deal more difficult than that reported in the research to date. This was found to be the case as shall be seen.

To assist in the understanding, a simple example is now given. Table 2.3 shows the operations involved for an exponent of $e = 1011_2$ (or 11_{10}) using the **exp_always_mul()** function. The inputs are already assumed to be in residue form (i.e $b \overset{residue}{\longmapsto} bR \bmod m =$ b_res and $1 \overset{residue}{\longmapsto} R \mod m =$ C_ans_res).

Next_ans	Shift	Decision - Ans	C_ans_res	Square
			R	bR
$R \overset{Mont}{\times} bR = bR$	$e^{\cdots 1011_2}$	Is $e_0 = 1$? - Yes	Update $\rightarrow bR$	b^2R
$bR \overset{Mont}{\times} b^2R = b^3R$	$e^{\cdots 101_2}$	Is $e_1 = 1$? - Yes	Update $\rightarrow b^3R$	b^4R
$b^3R \overset{Mont}{\times} b^4R = b^7R$	$e^{\cdots 10_2}$	Is $e_2 = 1$? - No	(No Update) b^3R	b^8R
$b^3R \overset{Mont}{\times} b^8R = b^{11}R$	$e^{\cdots 1_2}$	Is $e_3 = 1$? - Yes	Update $\rightarrow b^{11}R$	$b^{16}R$
	$e^{\cdots}$		Final Ans - $b^{11}R$	

Table 2.3: Example of modular exponentiation on the DUT with $e = 1011_2$ (all values are modulo m). The final answer is converted back into the normal domain by perfoming a Montgomery reduction on it: $b^{11}R \mod m \overset{reduction}{\longmapsto} b^{11} \bmod m$.

The **Square** and **Next_ans** columns are implemented in every iteration regardless of

[21] Signals that are in synchronisation with the clock but contain no information about the secret parameters and cannot be reduced by averaging.

the value of the exponent. After each bit has been operated on, it is shifted out of the register holding it and discarded (hence the **Shift** column). Once the last bit has been processed, and the processor is aware that the only bits left are zeros (this is determined using an array of OR gates), an interrupt occurs and the processor gets ready for another operation. As can be seen from the table, an update occurs only if bit $e[n] = 1$ and nothing happens if $e[n] = 0$. The value of C_ans_res is stored in a register of some sort. If updating this register produces a noticeable power fluctuation, then the system could be compromised by an attacker.

As mentioned, the difficulty in attacking this particular implementation comes from the fact that the multiplication is carried out in every round regardless of the value of the exponent. Most smart cards do not implement it this way as it requires more processing than necessary. However, due to parallelism, it does not have any adverse effects on the efficiency of the implementation studied here. If the multiplication was only conditionally executed (as in **exp_cond_mul()**), then the difference in power fluctuations between a 1 and a 0 would be far easier to recognise than that of a simple register update. The latter will tend to draw a lot less power and therefore be much more difficult to see. In fact, one of the suggested countermeasures against these attacks is to implement this multiplication with every iteration[22].

2.5.3 Points of occurrence of the fluctuations

To understand when these fluctuations should occur, either an SPA attack needs to be carried out or some extra information about the system be known. For this experiment, some extra information was known and as will be seen, was verified experimentally. The length used for the base and the modulus was $N = 2,174$ bits and the key was $2,176$ bits. The update of C_ans_res occurred every $2 \times (N + 2) + 1 = 2 \times 2176 + 1 =$

[22] This of course is not a perfect solution but it does create a lot more work for the attacker.

4,353 clock cycles due to the way the Montgomery's method was implemented[23]. The clock for this particular prototype ran at 60MHz which gave a minimum time between fluctuations as

$$\frac{1}{60e^6} \times 4353 = 72.55\mu s,$$

depending on whether or not two ones occurred consecutively with each other. Other timing characteristics noticed were integer multiples of this base unit. This minimum time of occurrence gave a maximum fundamental frequency of $f = \frac{1}{T} \simeq 13.8$ kHz (again depending on the key). It might be considered that in order to see these fluctuations the power need only have been sampled at a little over twice this maximum (due to the Nyquist theorem). However, this was not the case. The operation of updating the C_ans_res with Next_ans_res occurred within one clock cycle. This means that the fluctuation lasted approximately

$$\frac{1}{60e^6} \simeq 16.7ns.$$

It was initially thought that a sampling period of half this value would completely acquire the signal however this was not the case due to excessive noise and the fact that the averaged waveforms were very prone to misalignment (as a result of unstable triggering etc.). If only two samples were taken per clock cycle then a slight misalignment of the waveforms would produce incorrect results in the averaging. The sample rate used was therefore 2GSs^{-1} which gave about 33 samples per clock cycle.

It is generally accepted that the sample rate be at least 20 times the clock frequency. One report [34] in which an SPA attack was carried out on a PIC microcontroller (similar to the devices used in some smart cards) running at a frequency of 4 MHz, used a sampling rate of 200MSs^{-1}. This corresponded to 50 sample points per clock cycle

[23] Although the update time appears to have been dependent on the key, it in fact wasn't. If the key was three bits long the operations still took 4,353 clock cycles to complete.

which was more than adequate for that particular situation. Although the results were quite impressive, the operations running on the smartcard were minimal which reduced noise to a very manageable level. This would not be the case on a fully functional high speed cryptosystem such as the one being tested here. This particular board generally has 10 operations running at once on 10 different ALUs!

The next section looks at one of three attacks which can be used to exploit these power fluctuations. The particular attack examined in fact uses more than just the single power fluctuation to compromise a system but in fact exploits the fact that intermediate values for two different keys will be different also. This gives this particular attack more power than one that would exploit only the outcome of the conditional statement.

2.5.4 Attacking the exponentiation implementation

There are three main attacks on public key systems described in the literature and are

1. Single-Exponent, Multiple-Data (SEMD)
2. Multiple-Exponent, Single-Data (MESD)
3. Zero-Exponent, Multiple-Data (ZEMD)

The names of each describe the information available to the attacker. To implement the SEMD attack, the attacker needs to be able to run the encryption algorithm on a random number of plaintexts where only one known exponent is required. To implement the MESD attack, the attacker needs to have the ability to encrypt a single piece of data with the key being attacked and another chosen key. The ZEMD requires only that the attacker encrypt a random set of data values.

The SEMD and the ZEMD attack will not be described here as they were unable to be implemented in this particular research due to the access that was given to the

cryptographic board (more will be said about this in chapter 4). However, it must be pointed out that the research to date shows that the SEMD attack is the least powerful of the three and it is highly unlikely that the cryptosystem under review would be vulnerable to this attack due the extremely large amount of power waveforms needed to attack the exponent. The MESD attack was the only attack performed on in this instance due to restriction with the setup. It will therefore be described next.

2.5.4.1 Multiple-Exponent, Single-Data (MESD) attack

It might be expected that all an adversary has to do to attack the algorithm of figure 2.7 is to subtract an averaged power waveform obtained using an unknown exponent, from one obtained using a known exponent, such as one where every bit is a one. This should then leave no bias spikes where the conditional statement was executed and bias spikes where it wasn't; fortunately for the cryptosystem, it is not as straightforward as this. The intermediate results would generally be different for each key, and would cause such widely varying power fluctuations that it would be impossible to pick out the bias spike(s) of the signal of interest. However, if two exponents are exactly the same up to a certain bit position, then for a particular base b, the intermediate value C_ans_res of each, will be the exact same up to that point. The MESD attack exploits this fact to allow an attacker to determine a private exponent from a known exponent.

The attack begins by attacking the LSB of the unknown exponent and works it's way up to the MSB one bit at a time. To attack each bit, a chosen value b is exponentiated J times with the unknown exponent e_u. The averaged power waveform $awfu_k$ is then calculated. Next, a random guess e_g of this exponent is taken and is used to exponentiate b, J times as before. Depending on whether the bit of e_g being attacked is a 1 or a 0 will determine whether the averaged waveform is $awf1_k$ or $awf0_k$.

The attacker then computes the differential traces $\Delta D1_k$ and $\Delta D0_k$

$$\Delta D1_k = awfu_k - awf1_k \tag{2.20}$$

and

$$\Delta D0_k = awfu_k - awf0_k. \tag{2.21}$$

Initially the LSB is attacked as mentioned. Assuming this is the same for both e_g and e_u, then the intermediate calculated values should remain the same up to the conditional statement in the second iteration of the FOR loop. The power fluctuations will therefore be the exact same also and the differential trace should be 0 up to that point. After this point, widely varying fluctuations should appear. If however, the LSB of e_g and e_u are not the same, the intermediate values will be different and the differential trace will not be zero for any particular length of time at the beginning.

Once the LSB is known, the same technique can be used to attack the remaining bits of the exponent. To attack the nth bit, the section of the key discovered by previous runs of the attack are now used as the first $n-1$ bits of e_g. By obtaining the average waveforms $awf1$ and $awf0$ (in which the nth bit is guessed to be either a 1 or a 0 as before), the differential traces $\Delta D1_k$ and $\Delta D0_k$ can be obtained. As before, the values of the intermediate results up to the conditional statement of the nth loop should be the exact same (assuming the key was guessed correctly up to this point). The differential trace should therefore equal zero up to this point and then widely varying fluctuations from there on.

This attack has been shown in [27] to succeed against smart cards with as little as 100 exponentiations per bit. This is certainly not the case for the DUT here as will be seen.

2.6 Summary

This chapter introduced a very powerful side channel attack known as power analysis. It has been an active area of research since it's introduction in 1998 by Paul Kocher et al. Smart card vendors have been particularly worried as numerous smart cards have been found to be highly susceptible to it. The initial area looked at was Simple Power Analysis (SPA). This involves gaining knowledge about the cryptosystem and it's algorithm by visually analysing a single power waveform. Although generally not a direct threat, it can offer assistance in other attacks.

Differential Power Analysis (DPA) was then discussed in detail. This attack begins by capturing a large number of power waveforms. These waveforms are then partitioned into two sets according to the output of a partitioning function that is dependent on a number of values, one of which is a subset of the key. The other values are known to the attacker and the subkey is guessed at. Each partition is then averaged to reduce noise and the averages subtracted to produce a differential trace. This procedure is repeated until every possible value of the key has been guessed at. For DES the particular subkey attacked is only six bits long so this implies there will be $2^6 = 64$ differential traces. The one showing the largest bias spikes is more than likely the trace produced by the correct guess at the subkey. The remaining subkeys can then be revealed by attacking the other s-boxes and the final eight bits by an exhaustive search.

Improvements of the basic differential power attack were then discussed and included muliple-bit DPA and high order DPA. This then led onto attacks on public key systems and specifically, modular exponentiation. The main attack discussed was the Multiple Exponent Single Data attack (MESD) developed by Messerges et al. This is a very powerful attack an was the one used to attack the DUT in this research. It begins by taking a guess at the first bit of the key and measuring multiple power waveforms for both the unknown and guessed keys. Each set of waveforms is averaged and the

resulting two waveform subtracted as in normal DPA. If the guess was correct then the differential trace produced should show a value of zero up to the point where the next bit is manipulated. However, if it was incorrect then the trace should not go to zero. This technique is then repeated for each bit until the entire key is found.

Power attacks are very well understood and mitigation techniques are constantly being devised to protect cryptographic devices against them. One idea is to provide some sort of tamper resistance facility that will detect any physical attempt at accessing the device. Using these and other techniques, risks of power analysis attacks may be reduced significantly. The attacker may then try to find some other way to gain secret information without making any physical contact with the cryptosystem itself. This will then render the tamper sensing circuitry useless. It has been discovered that this can be achieved with the use of antennae to capture the electromagnetic radiation being emitted from the device. The name given to this attack is *Electromagnetic Analysis* (EMA) and has been known about in military circles for quite some time. The US Government has developed the TEMPEST program to try and deal with these compromising emanations. Both EMA and and TEMPEST will be discussed in the next chapter.

Chapter 3

Electromagnetic Analysis (EMA)

The last chapter looked at the ways in which the power consumed by a device can be used to extract information about the operations it is running. This power results from the voltage and current that the device uses to operate. If the voltage is considered constant (as a result of regulation) then the current will fluctuate in direct proportion to the power[1]. This fluctuating current will generate a sympathetically fluctuating magnetic field in it's vicinity which in turn will create a sympathetically fluctuating electric field. If these fields can be captured using an antenna of some sort, then it is very likely that techniques similar to those in the last chapter can be used to extract secret information. The name given to this form of attack is *Electromagnetic Analysis* (EMA) [41].

Governments and military organisations have been aware of the problem of compromising radiated emissions for quite some time and a program known as TEMPEST has been set up to try to deal with it[2]. Millions of dollars have been directed at the TEMPEST program and nearly all of its findings have been classified. As a result there is little information in the public domain. Fortunately, academic research in this area has been on the increase in the last couple of years and a number of papers can be found (some examples being [42,43]).

It is the aim of this research to determine whether a certain cryptographic accelerator is

[1] This is an over simplification of the problem but serves for the purposes of an introduction.

[2] A number of sources state that TEMPEST deals with the reduction of *all forms of compromising* emissions and not just those that are radiated. It has generally been used in the public domain to refer to radiated emissions only, and for the purpose of this thesis will refer only to this form of emission also.

vulnerable to EMA and power analysis and to allow designers of cryptosystems in general to use the setup here to test their own products for vulnerabilities. It is important therefore to understand the theory behind the generation and capture of electromagnetic radiation. This chapter will therefore begin by discussing some of this theory which although not exhaustive, should act as good introduction. Models will be presented that allow an estimation of the level of radiation emanating from a circuit which can then be used to determine the locations and frequencies that are most compromising. Following this, a brief discussion of the TEMPEST program will be given. The techniques used here to test the DUT are not claimed to be the same ones used by professional TEMPEST engineers when testing equipment, but they do act as a starting point for designers wishing to test their systems against these attacks. A literary survey of all the relevant publications on the subject of EMA will then be reviewed and it will be seen that a lot of work still remains to be done in this area.

3.1 Electromagnetic Radiation of Electronic Systems

In EMC it is taught that for interference to be a problem, three components are required: An emitter, a coupling channel and a receiver. This idea can be interpreted to include an EMA attack where the receiver is the antenna of the adversary, and the emitter is the cryptosystem under attack. Elimination of any one of the above elements will eliminate the compromising emissions. This can mean a (re)design of the cryptosystem for minimum emissions. Generally nothing can be done about the receiver, however the coupling channel can be reduced as much as possible to stop it from picking up the emissions.

Any setup within the cryptosystem that acts as an efficient radiator of electromagnetic energy needs to be restructured[3]. As well as this, any coupling channel within the

[3]Generally it will not be possible to eliminate the radiation altogether.

device must somehow be severed or blocked [44]. For radiated emissions the obvious choice would seem to be that of shielding, however this is more of a remedy than a cure and other strategies should be considered initially. It must be kept in mind that the sensitive information may travel to another part of the circuit by conduction and re-radiate at unexpected locations. As a result, shielding in one location may not offer as much protection as might be expected.

To reduce the radiation properties of a system it is necessary to understand the characteristics that cause it to radiate. These same principles can be used to understand what is required to capture this radiation and will assist in testing for vulnerabilities. With the knowledge gained from the theory and the testing, mitigation techniques can be deployed to reduce potential threats.

3.1.1 Basic Concepts

Radiation from electronic systems is generally classified under two headings: *differential mode radiation* and *common mode radiation*. Differential mode radiation is due to the currents that are setup as a result of normal circuit operation. These currents flow around the loop enclosed by the signal wires and are in opposite directions to each other. Differential mode radiation is the easiest to eliminate as it can be controlled by careful layout of the components making up the circuit.

Common mode radiation on the other hand would not be part of the circuit design and is due to grounded sections of the circuit being at a potential different from true ground. This is usually to do with unintentional impedances of the Printed Circuit Board (PCB) traces which cause undesired voltage drops. The radiation problem then occurs when (for example) a cable of some sort is connected to the PCB. The shield of the cable will be connected to the "ground" of the circuit which may not be at true ground. The current flowing through the impedance from the "ground" connection

to true ground may fluctuate in accordance with the operations on the board[4]. The fluctuating voltage created across the impedance will cause the shield of the cable to fluctuate in a similar manner. The cable being quite long will act as a fairly efficient radiator of electromagnetic energy and transmit the compromising information.

To understand the radiation effects more clearly, it is necessary to have some sort of a model that can be used to describe and calculate the radiation being emitted from a circuit. Maxwell's equations are normally used to solve problems involving electromagnetic quantities however the complexity of the circuits involved make this an extremely difficult (if not impossible) task. With some simplifications it is possible to develop a model that will give an order of magnitude of the radiated emissions. This can be used to determine the emission levels for a certain device. It is generally the worst case scenario that is taken for each situation encountered. An attacker may use this strategy to determine the strongest radiating component of a device and use it to reduce the length of time necessary for an attack.

3.1.2 Modeling radiation from circuits

Each radiation mode can be modeled separately. Differential mode radiation can initially be thought of as being emitted from an ideal zero impedance loop. However, common mode radiation is generally modeled as a high impedance straight wire (i.e. a monopole antenna). For both cases, the fields generated are dependent on the characteristic impedance of the source, the media surrounding it and the observation distance. Although alternating electric and magnetic fields cannot exist independently of each other, the nature of these fields and the relationship they have with each other, changes significantly as the distance from the source is increased. As a result, the area around

[4]This is similar to the idea of placing a small resistance in series with the power or ground line in order to measure the power fluctuations (see section 2.3).

the source is generally split into the following sections[5] (found using Maxwell's equations):

- The Near Field - this extends from the source to about $r = \lambda/2\pi$ (λ is the wavelength of the source voltage/current).
- The Transition Region - this is the area around $r = \lambda/2\pi$ and extends to about $r = 1.6\lambda$.
- The Far Field - this technically begins at $r = \lambda/2\pi$ but is not completely in effect until about $r = 1.6\lambda$.

In the near field, the fields are predominantly reactive and are therefore known as *reactive fields*. These can be either inductive (if due to a low impedance, high current structure) or capacitive (if due to a high impedance, low current structure). In the far field however, the fields are predominantly radiated and are therefore known as *radiated fields*[6]. The waves are considered plane waves at this point and beyond and are only inversely proportional to the distance from the device, i.e. $\propto 1/r$ as opposed to $\propto 1/r^3$ (which is the case for the inductive field as shown in table 3.1 below). In the transition region the non-radiated (reactive) fields begin to drop off and the radiated field begins to dominate, however a combination of both still exist.

Taking measurements in each of these regions will have different consequences. If an antenna is used to measure the energy in the near field, then its presence will affect the measurement. This is due to the fact that the fields at this point are reactive and are storing energy which under normal circuit operation is fully returned to the source on each half cycle. However, an antenna placed in the reactive field will absorb some

[5]It must be pointed out that the definitions given assume that the distance r from the radiated body is large compared to it's dimensions. If it is not then other effects must be taken into consideration.

[6]There is a common misconception that the reactive field is also radiated - this is not the case. The ambiguity probably stems from the fact that both terms are sub-headings under the more general "radiated emissions" section of EMC terminology.

of this stored energy and prevent it from returning to it's source. This causes energy losses in the originating circuit and affects the measurement (a similar problem exists when connecting a finite impedance scope probe to a node in the circuit). If the antenna absorbs too much energy it can end up loading the source. In the far field, the field is not reactive and the energy is not stored but is propagating away from the source. Any measurement of the radiated field will have no effect on the source as this energy will not be returned anyway. These points need to be considered when testing systems for vulnerabilities to EMA.

As mentioned already, the equations used to describe these fields can become quite complicated and the fact that the characteristics of the fields themselves change only increases the complexity of the situation. However with some simplifications, a good estimation of the field levels can be achieved. For the loop and the wire model, the following is therefore assumed:

- The current I is uniform in the loop or length of wire.
- The circumference of the loop and the length of the wire are $\ll \lambda$ and r (the wavelength and point of measurement).
- Both are in free space and are not close to a conducting surface.
- The loop has no impedance other than it's own reactance.

Using these assumptions, the field equations can be greatly simplified. These equations are developed in [45] and are summarised in table 3.1 where the worst case values are given. The "ratio" column shows the ratio of the electric field to the magnetic field (i.e. E/H). This is known as the *wave impedance* and in the far field is equal to the impedance of the medium through which the energy is traveling[7]. For free space this

[7]For the purposes of this discussion the medium will be free space.

is given by

$$Z_o = \sqrt{\frac{\mu_o}{\epsilon_o}} = 120\pi \approx 377\ \Omega. \tag{3.1}$$

where $\mu_o = 4\pi \times 10^{-7}\ \mathrm{m}^{-1}$ is the *permeability of free space*, $\epsilon_o = 1/36\pi \times 10^{9}\ \mathrm{Fm}^{-1}$ is the *permittivity of free space* and Z_o is the *free space impedance*. The other values shown are as follows:

- I is the peak-to-peak loop current (A)
- A is the area of the loop (m^2)
- λ is the wavelength (m)
- r is the observation distance (m)
- Z_o is the free space impedance (Ω)

Configuration	**Region**	**Electric Field** ($\mathrm{E}_{V/m}$)	**Magnetic** ($\mathrm{H}_{A/m}$)	**Ratio E/H** (Ω)
Loop	Near Field	$Z_o IA/2\lambda r^2$	$IA/4\pi r^3$	$Z_o(2\pi r/\lambda)$
	Far Field	$Z_o IA\pi/\lambda^2 r$	$IA\pi/\lambda^2 r$	Z_o
Straight Wire	Near Field	$Z_o Il\lambda/8\pi^2 r^3$	$Il/4\pi r^2$	$Z_o(\lambda/2\pi r)$
	Far Field	$Z_o Il/2\lambda r$	$Il/2\lambda r$	Z_o

Table 3.1: Equations for the fields produced by an ideal loop and wire structure in both the near and far field. The ratio E/H is known as the wave impedance and is equal to the free space impedance in the far field.

A number of points should be noted with regard to table 3.1.

1. The magnetic field of the loop in the near field is dominant when r is small and λ is large. The opposite is true for the straight wire antenna where the electric field

dominates. This is generally why the loop antenna is associated with magnetic fields and the monopole with electric fields. It does not mean that the other field is not produced but only that it is the least dominant of the two.

2. For the same reasons the wave impedance is small for the loop antenna and large for the monopole antenna. It should be noted then that the wave impedance in the near field is related to the source that produced it.

3. The magnetic field of the loop in the near field is proportional to $1/r^3$ whereas the electric field is proportional to $1/r^2$. The opposite is true for the straight wire antenna. Separation of a receiver from the source in the near field has more of an effect than in the far field (where the fields are proportional to $1/r$).

4. In the far field the electric field is always larger than the magnetic field by a factor of $Z_o = 377\Omega$ which is the impedance of free space. Knowing one therefore allows calculation of the other.

5. The magnitude of the radiation from the loop is always dependent on it's area A. This is a very important point as it shows that the loop area of a circuit should be kept as small as possible (for example by keeping signal leads as close as possible to their returns).

6. Ott shows the equation for the electric field from loop antenna as being equal to [46]

$$E = 131.6 \times 10^{-16}(f^2AI)(\frac{1}{r})sin\theta \tag{3.2}$$

in free space, far from a conduction surface (such as a ground plane) and

$$E = 263 \times 10^{-16}(f^2AI)(\frac{1}{r})sin\theta \tag{3.3}$$

when placed over a ground place. This is a 6dB increase in emission levels

(again a worst case result) and must be taken into account when trying to reduce radiated emissions. Equation 3.2 is the exact same as that shown in table 3.1 for the electric field of a loop antenna measured in the far field[8]. This equation and that of a straight wire should therefore be doubled when in proximity to a ground plane. The latter will therefore be

$$E = \frac{2Z_o Il}{2\lambda r} \approx \frac{12.6 \times 10^{-7} fIl}{r} \tag{3.4}$$

Unfortunately, the ideal loop and wire are not always the best models from a practical point of view and a modification needs to be taken into account. This more practical circuit is shown in figure 3.1 where l is it's length and w is it's width. The equations used for this model are determined by the total circuit impedance ($Z_{total} = Z_{wire} + Z_{load}$). If this impedance is less than $7.9 \times r \times F$ (where r is in meters and F is in megahertz) then the ideal loop formulas are used. However, if it is greater than this value then a modified straight wire model is used. A summary of the resulting equations (developed in [45]) is given in table 3.2.

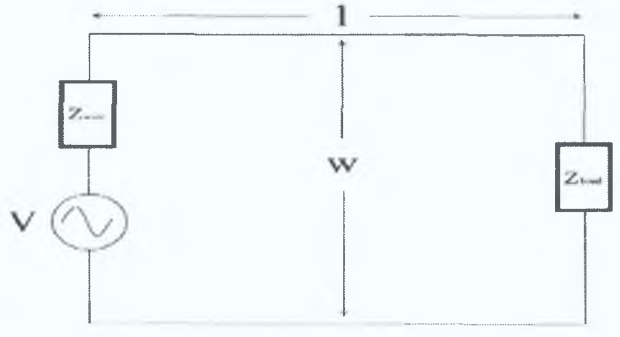


Figure 3.1: More practical model of a radiating circuit. The total impedance of the circuit is $Z_{total} = Z_{wire} + Z_{load}$ where Z_{wire} is the impedance of the wire and Z_{load} is the total impedance of any loads driven by the voltage.

[8]This can be seen by substituting $Z_o = 377\Omega$, $\lambda = c/f$ (where $c \approx 3 \times 10^8$) and setting $\theta = \pi/2$ (worst case).

The voltage (V) is used instead of a current due to the fact that the former is more easily determined. The units used are different in order to produce more practical results and are as follows:

- V is the peak-to-peak driving voltage (V)
- A is the area of the circuit (cm^2)
- F is the frequency (MHz)
- r is the observation distance (m)

It can be seen from these equations that the magnitude of the radiation in all cases is proportional to the area of the circuit. Also, the radiated field is proportional to the square of the frequency (for differential mode signals) and the frequency (for common mode signals)[9] which is why higher clock frequencies are a major problem in both EMC and security applications.

	$\mathbf{Z}_c$	**Electric Field** ($E_{\mu V/m}$)	**Magnetic** ($H_{\mu A/m}$)
Near Field	$< 7.9 \times r \times F$	$63IAF/r^2$	$7.96IA/r^3$
	$> 7.9 \times r \times F$	$7.96VA/r^3$	$7.96IA/r^3$
Far Field	-	$1.3VAF^2/rZ_c$	$1.3VAF^2/rZ_oZ_c$

Table 3.2: Equations for the fields produced by the more practical model in both the near and far field. For the near field, the equations are dependent on the impedance of the circuit.

[9] Although the practical model implies that it is proportional to the square of the frequency for both cases this is dependent on a number of factors and common mode radiation is more generally proportional to the frequency (see [45] for more details).

3.1.3 Spectral content of signals

The equations shown in tables 3.1 and 3.2 were developed for waveforms of a single frequency (i.e. sinusoidal waveforms). It is rare that any signal (even intended ones) will be purely sinusoidal and most signals will be some combination of sinusoidal frequencies. It is therefore necessary to determine the spectral content of a signal in order to know how it will radiate. Most of the time this is an extremely complicated task and often a number of simplifications can be made.

As a practical example of a waveform consider the clock signal of a digital system. This signal is not a perfect square wave but is in fact a trapezoidal waveform and has finite rise and fall times (denoted by t_r and t_f respectively). It is also possible that the signal may not have a 50% duty cycle. A diagram of a typical digital signal is shown in figure 3.2.

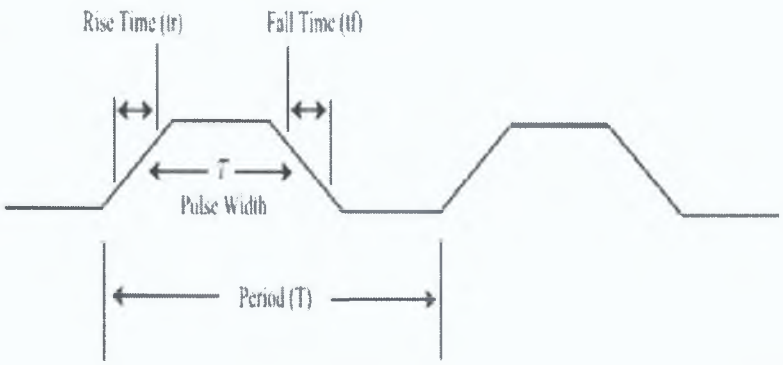


Figure 3.2: Trapezoidal waveform with finite rise (t_r) and fall (t_f) times, pulse width τ and period T. The duty cycle is $d = \tau/T$.

It is shown in [46] that the harmonic content of this signal is given by

$$I_n = 2Id\frac{sin(n\pi d)}{n\pi d}\frac{sin(n\pi t_r/T)}{n\pi t_r/T} \tag{3.5}$$

where I_n is the nth harmonic, I is the peak to peak amplitude of the current, d is the

duty cycle, t_r is the rise time (the smaller of the two if the rise and fall times aren't equal) and T is the period.

Figure 3.3 shows the value of the first 20 harmonics for a 50% duty cycle and 2ns rise time. As can be seen, only the odd harmonics are present and the fundamental frequency is 64% of the signal current. If the duty cycle is not 50% however, then even harmonics will also be present. In the case of an 80% duty cycle for example, the fundamental frequency will only be 37% of the signal and the second harmonic (which isn't present for a 50% duty cycle) contains 30% of the signal content. This can be seen in figure 3.4.

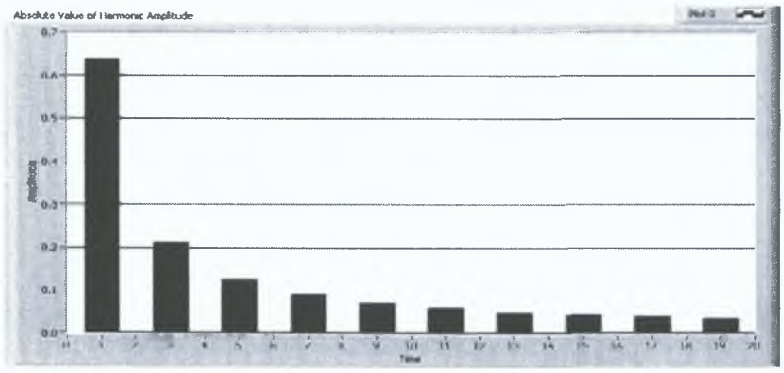


Figure 3.3: First 20 harmonics present in a square wave with 2ns rise time and 50% duty cycle. The fundamental frequency is .64 times the peak amplitude of the current.

It would be fairly tedious if every harmonic had to be calculated for every signal encountered. Luckily there is another method that is generally used. It is known as the *envelope method* and it shows the envelope of the peak values of the spectrum of the signal. It can used to estimate the level of radiation at a certain frequency.

Figure 3.5 shows the envelope of the square wave described by equation 3.5. As can be seen from this figure, the harmonics are constant up to the corner frequency

$$f_{c1} = \frac{1}{\pi\tau} \tag{3.6}$$

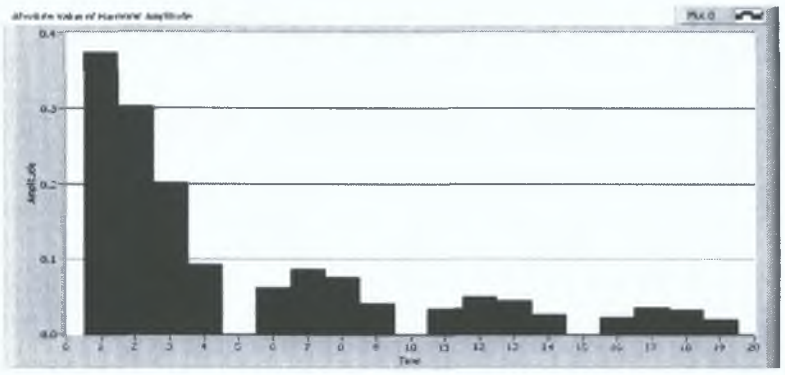


Figure 3.4: First 20 harmonics present in a square wave with 2ns rise time and 80% duty cycle. The fundamental frequency is now only .37 times the peak amplitude of the current and even harmonics are present. Every multiple of the 5th harmonic is zero however.

where τ is the pulse width of the waveform. From this point on, they drop off at 20dB per decade up to next corner frequency

$$f_{c2} = \frac{1}{\pi t_r} \tag{3.7}$$

where t_r is the rise time. After this point they drop of at a rate of 40 dB per decade. It can be seen therefore that the shorter the rise time, the higher the spectral occupancy.

As technology improves and rise times get shorter, the energy of the higher harmonics will increase. Because far field radiation increases as the square of the frequency (for differential mode signals) and frequency (for common mode signals), the fields emanating from a device will occupy increased levels of bandwidth. This holds consequences for information security as well as EMC even though the signal of interest to an attacker will generally be at lower frequencies than the clock harmonics. If this signal happens to modulate the harmonics of the clock then it will have a method by which to radiate with increased intensity. A number of researchers have shown this to be a problem (as shown later) and in particular Agrawal et al. present two graphs showing the modulated signal (which clearly shows increased levels of compromising

activity) and it's demodulated baseband [32]. Modulated signals are a very attractive source of information as they allow lower frequency signals to propagate far from the source.

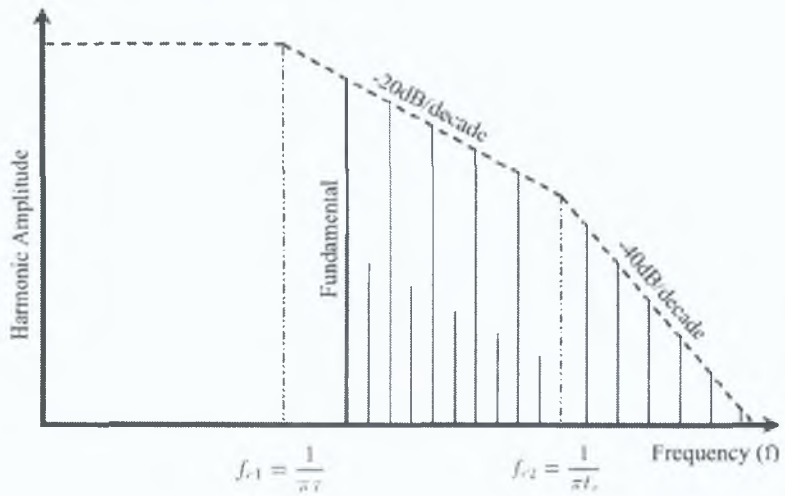


Figure 3.5: Envelope of the spectrum of the trapezoidal wave shown in figure 3.2.

Figure 3.5 shows the envelope of the spectrum of a trapezoidal waveform. It can be seen that the fundamental frequency f_1 is greater than the corner frequency f_{c1}. Because of this, the harmonics start decreasing at a rate of 20dB per decade. However, this may not always be the case. If the fundamental falls below f_{c1} then they will remain constant up to that point. The fundamental frequency will equal f_{c1} if

$$\begin{aligned} f_1 = \frac{1}{T} = f_{c1} = \frac{1}{\pi\tau} \\ \Rightarrow \tau = \frac{T}{\pi} \end{aligned} \tag{3.8}$$

which implies that the fundamental frequency will fall below the first corner frequency if

$$\frac{T}{\pi} > \tau > 0 \tag{3.9}$$

and will fall above it if

$$\frac{T}{\pi} < \tau < T. \tag{3.10}$$

Each situation will produce a slightly different envelope.

As mentioned on a number of occasions, the radiated emissions from a circuit is proportional to the square of the frequency for differential mode signals and to the frequency for common mode signals . This is equivalent to saying that the emissions *increase* at a rate of 40dB/decade and 20dB/decade respectively. However, the amplitude of the harmonics of the signal in figure 3.2 *decrease* at a rate of 20dB/decade up to f_{c2} and at a rate of 40dB/decade from then on. As a result, the radiated portion of the differential mode signal will be constant up to f_{c2} and will decrease at a rate of 20dB/decade from this point on. For the common mode signal it will be constant up to f_{c_2} and will decrease at 20dB/decade from then on. This is depicted in figures 3.6 and 3.7 respectively[10].

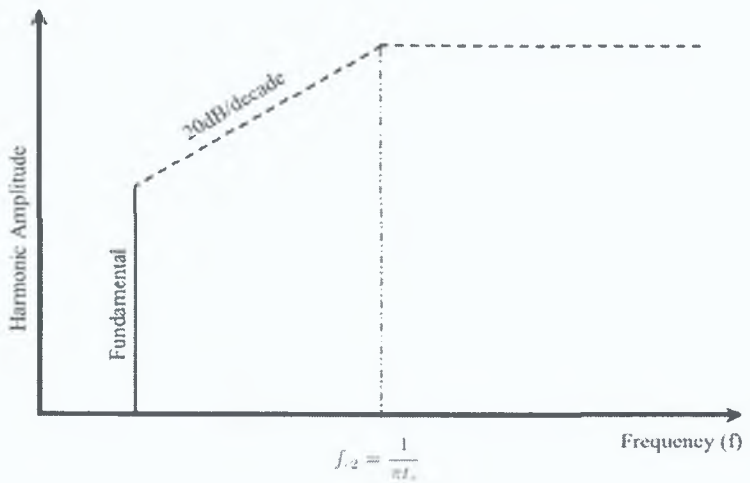


Figure 3.6: Differential mode radiated signal.

Having looked at some of the radiation properties of circuits, the next step is to de-

[10]For simplicity each of these graphs assume that $\frac{T}{\pi} < \tau < T$.

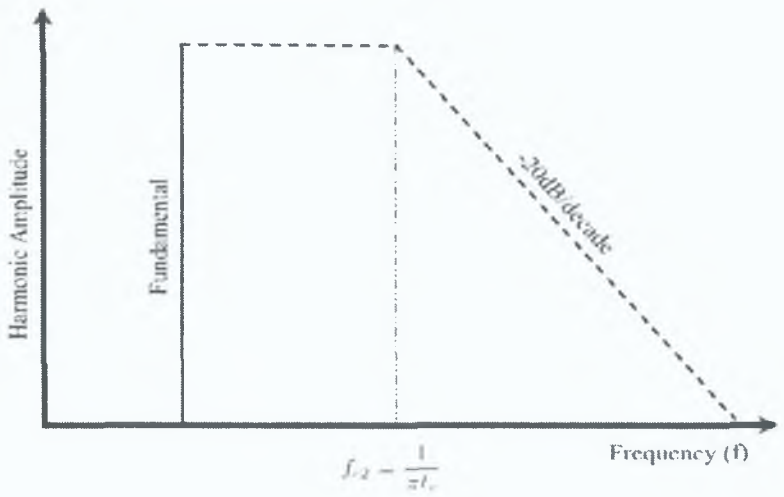


Figure 3.7: Common mode radiated signal.

termine the state of the art in using these emissions for attacks. TEMPEST being the oldest body of knowledge will be discussed first. As mentioned already, information on TEMPEST is extremely scarce and most of it is pure conjecture due to the fact that it is classified. However, it is interesting to have a brief look at what is available. Following this the publicly available literature will be discussed. As mentioned, this area is still quite new and a lot more research needs to be conducted.

3.2 TEMPEST

Phrases like "TEMPEST attacks" are commonly found in literature to refer to the use of electromagnetic analysis in attacking information systems. However, TEMPEST is not an actual attack itself. According to the available literature, TEMPEST is an unclassified word for a classified program. It has been run by the United States government[11] since around the late 50's early 60's[12], although the idea of compromising

[11] From here on in the term "Government" will be used to refer to the United States government unless otherwise stated.

[12] This is speculation of course but it is the most generally accepted time period available from public sources.

radiation was known long before this date (a time-line of TEMPEST can be found at [47]). There are a number of suggested meanings for this word in the public domain including the suggestion that it is an anagram and stands for (to mention the more popular of a number of variants) - **T***ransient* **E***lectro***M***agnetic* **P***ulse* **E***manation* **ST***andard*[13]. It is more likely however, that TEMPEST is not an anagram, but is simply a word used to refer to the area of research that deals with protecting equipment processing sensitive information from being extracted through radiated emissions. This radiation may be produced without any outside interference (i.e. a passive attack) or may be generated by external means such as shining a threat frequency onto a device in order to cause the circuit to resonate thus emitting stronger radiation (i.e. an active attack).

Bruce Gabrielson attempts to explain what TEMPEST is in [48] where he provides his own definition of the word. From his paper (and other publicly available sources), it is likely that TEMPEST deals with the other forms of compromising emanations but this is unclear. This can be seen from the following four definitions of TEMPEST each taken from a different source. Definitions one and two differ slightly from three and four:

1. "**TEMPEST:** Military code-name for activities related to van Eck monitoring[14], and technology to defend against such monitoring". [49]

2. "**TEMPEST** is a U.S. government code word that identifies a classified set of standards for limiting electric or electromagnetic radiation emanations from electronic equipment". [50]

3. "**TEMPEST**: (a). [A] Short name referring to investigation, study, and control of compromising emanations from information systems (IS) equipment. (b). To

[13]This probably stems from the fact that EMP is an anagram for Electromagnetic Pulse.
[14]See section 3.3.

shield against compromising emanations". [12]

4. "**TEMPEST**: An Unclassified short name referring to investigations and studies of compromising emanations. It is sometimes used synonymously for the term 'compromising emanations': For example, TEMPEST tests, TEMPEST inspection". [51]

From these definitions it appears that TEMPEST is not the proper name of the program run by the Government and probably reveals no information about it (due to the fact that it's unclassified).

There are a number of documents available to the public which have been released as a result of the *Freedom Of Information Act* (FOIA). A lot of these can be found at [52] and are a good starting point for developing an understanding of the TEMPEST program and it's findings. It must be said however that the documents found here are highly redacted, especially in the areas of interest. A good example of this can be seen in [53] which states three areas of risk for cryptosystems - low, medium and high - but the definitions for these have been almost completely replaced by x's. Some interesting but non-technical documents can be found at [50,54] and some slightly more technical ones at [55–57].

It is explicitly stated in more than one of these documents that facilities processing unclassified information will not have TEMPEST countermeasures applied to them. This would suggest mainly that TEMPEST countermeasures are costly and/or time-consuming. Potts confirms this in [58]:

> "Commercially there is little need for equipment to be shielded to military TEMPEST standards, both from the point of view of cost as well as level of attenuation. TEMPEST standards effectively provide full shielding and are commercial overkill in this area".

As a result of this high cost, a system must first go through an inspection of a minimum of six key areas before mitigation techniques are implemented. These are (a more detailed description can be found in [53]):

1. Location
2. Volume of Information Processed
3. Sensitivity of Information Processed
4. Perishability of Information Processed
5. Physical Control
6. TEMPEST Profile of Equipment

On completion of these tests, the *Certified TEMPEST Technical Authority* (CTTA)[15] decides whether or not the equipment needs TEMPEST protection and if so to what level. Devices that are fully approved under the TEMPEST standard are placed on the Preferred Products List (PPL). This document lists all approved TEMPEST equipment, consultants, test houses and shielding devices and is maintained by the Government administrator of the Industrial TEMPEST Program (ITP) [59]. Any product on this list will have met the *NACSIM 5100A* specification (the NATO equivalent of this test standard is the *AMSG 720B*). This is a classified document and is unfortunately not available for public viewing.

To ensure that protected equipment are not effected in any way by unprotected equipment through coupling etc., two general equipment categories have been identified[16]: RED and BLACK. The following definition [12] gives a general idea of what these terms mean:

[15]This is a Government employee that is experienced and technically qualified in the area of TEMPEST.

[16]This terminology can be used for systems at any level of abstraction.

> "**RED/BLACK concept:** [The] Separation of electrical and electronic circuits, components, equipment, and systems that handle national security information (RED), in electrical form, from those that handle non-national security information (BLACK) in the same form".

It is clear from this that the processing of classified and non-classified signals in the same system/area is not an option under the TEMPEST program. This way radiation given off by a RED system will not be coupled onto a BLACK system. This is clearly a standard practice and should be a consideration of the designers of cryptosystems. The three mechanisms outlined in section 1.2.1 (i.e. conducted, radiated and reactive coupling) are justification enough for the Government to physically and electrically isolate equipment and to classify different systems into their RED and BLACK categories.

3.3 Literary Review

Although the idea of compromising radiated emissions has been known by Government and military officials for almost 90 years[17], the general public were not aware of it until around 1985. It was at this time that the Dutch engineer Wim van Eck published a paper reporting an experiment that used unintended radiated emissions to reconstruct potentially sensitive data. His results caused quite a stir throughout the security community.

Van Eck's paper [60] discusses the idea that electromagnetic radiation emitted from the Cathode Ray Tube (CRT) of a Visual Display Unit (VDU) can be used to reconstruct the screen's contents. Although van Eck claims it is possible to reconstruct a signal up to a distance of 1km (or more) this is debatable[18]. It is clear however that success at relatively large distances can be achieved due to the fact that the video signals are

[17] According to [47] the first account of radiated emissions being used in this way was in 1914.
[18] Having said that, some good arguments are presented which may make it plausible.

amplified from Transistor-Transistor Logic (TTL) levels up to several hundred volts. The radiated signals can be collected and reconstructed with minimal, "off the shelf" equipment. In fact, van Eck used an ordinary television receiver to demodulate and display the captured information. The television of course had to be modified slightly to allow the synchronisation signals (which control the scanning electron beam) to be adjusted to the correct value (i.e. those used by the VDU being attacked) but this didn't appear to cause any great problems. As a result of van Eck's work, the public domain term for monitoring radiated emissions became known as *van Eck monitoring*. It became clear that a way was needed to predict the radiation not only in terms of EMC but also in terms of security. Ward discusses a number of options available to the designer for the prediction of radiated emissions in terms of security [61].

It is interesting to note that due to the furore caused by van Eck's paper, all subsequent research was classified by the management of van Eck's funding company [62]. However, the idea had already begun to spread throughout the research community and since then a number of papers have been published all of which verify the integrity of van Eck's work [57,63–70]. Although these papers are all very interesting, they do not answer the question of whether or not cryptographic systems such as smart cards and high speed cryptographic systems are at risk from the same form of eavesdropping. Initially it might appear not to be a problem as these devices use only a fraction of the voltage levels used by VDUs however the current research would seem to suggest otherwise.

Daneffel published a paper three years after van Eck stating that ISDN lines were also vulnerable to electromagnetic eavesdropping [71]. It was now clear that high voltage VDUs weren't the only systems at risk. The results showed that it was possible to reconstruct data with common loop and biconical antennas placed 10cm and 1m from the source, respectively. It is well known in the EMC community that one of the

strongest radiated signals in a digital system is that of the clock. Daneffel realised that there was a possibility that the clock signal could be modulated with the data:

> "However, clock signals may also be emitted via the [ISDN] lines... Due to nonlinear components, it is conceivable that these HF or VHF signals might be modulated by audio, video or digital signals in the ISDN terminals. Such information may be picked up and demodulated by special receivers, and the original information can be recovered".

This was a fairly significant result as it meant that lower frequency signals could be efficiently radiated by modulating the higher frequency clock signal. Using this idea, Daneffel tried a number of demodulators in an attempt to reconstruct the signal. Although it was not stated whether or not all of them worked, it was reported that the FM demodulator gave the best results. This suggested that the data signal could somehow change frequency of the clock signal although no explanation was given.

Results similar to Daneffels were reported in 1990 by Peter Smulders where he showed that RS-232 cables were also vulnerable to "van Eck monitoring" (as it was now known) [13]. Smulders claimed that the data could be picked up with the aid of a simple radio receiver. This was possible because the bit rates of the data signals were low compared with the Nyquist rate of the receiver. Figures 3.8 and 3.9 show some of the results reported by Smulders.

Figure 3.8 shows data that was sent down an RS-232 cable and that picked up by a radio receiver placed 7m away. The receiver was tuned to 16MHz and it's AM detector was used to reconstruct the data. The complete pulses are not visible in the received signal which (according to Smulders) is due to the AM envelope detection process. Figure 3.9 (also taken from [13]) shows the same data signal picked up with the receiver tuned to 98MHz using an FM detector. It can be seen in this case that the signal is almost

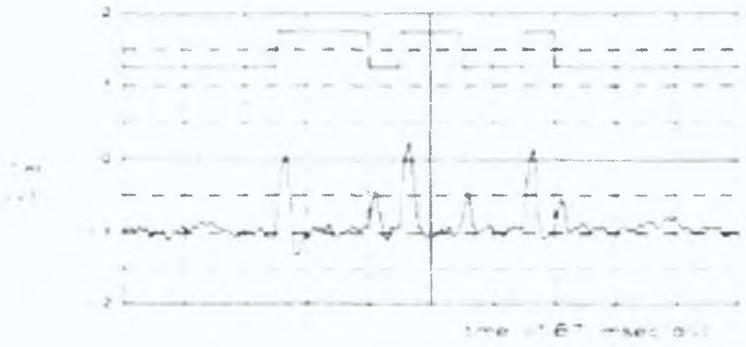


Figure 3.8: Graph presented by Smulders that compares the original data on an RS-232 cable with the intercepted data. All that was used was a simple pocket radio tuned to 16MHz and placed 7m from the source. At this frequency the receiver uses an AM detector which results in loss of the signs of the transitions.

perfectly reconstructed. This can be compared with Daneffels results (described above) where it was claimed that the FM demodulator produced the best results.

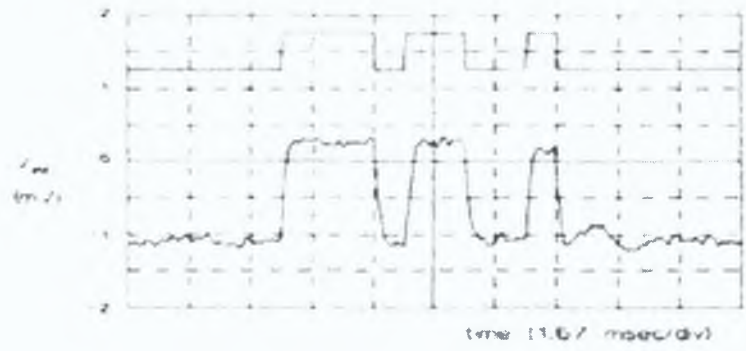


Figure 3.9: As with figure 3.8 a radio receiver placed at 7m from the source was used. This time however it was tuned to 98MHz and FM demodulation was used. The full signal is present as can be seen.

As with the VDU case, Daneffels and Smulders papers don't fully answer the question about the level of vulnerability of a modern day high speed cryptographic processors although they do suggest that a problem may exist. In the case of a cryptoprocessor there are other factors to consider which complicates the attack. For example, the data in the cryptoprocessor is not transferred between subsystems in serial format. As a

result, a lot of different bit changes would be picked up at once increasing the level of difficulty of the attack. Also due to it's highly parallel architecture, a lot of other operations occur at the same time as those of the key operations and hence cause a lot of noise on the waveforms detected. Using power analysis techniques however, it should be possible to reduce these complications considerably[19].

In 2000 Jean-Jacques Quisquater and David Samyde took Smulders and Danfelles research a step further and applied the idea to cryptographic smartcards. They presented their results at [11] where it was shown that the techniques used by Kocher in [10] could be modified to include van Eck monitoring. These modified attacks were referred to as *Simple Electromagnetic Analysis* (SEMA) and *Differential Electromagnetic Analysis* (DEMA) and collectively as *Electromagnetic Analysis* (EMA). These names are obviously variants of simple power analysis, differential power analysis and power analysis respectively.

The ideas presented at [11] were then later expanded upon in [72]. The ability of the SEMA method to determine the Hamming weight of a word of data was explored and was found to be possible. As before, this may lead to the possibility of a decreased effort brute force attack. It was also pointed out that even though smartcards are protected against many attacks using methods such as tamper resistance it is not possible to detect an EMA attack in this way.

Although Quisquater and Samyde showed that it was possible to pick up these radiated emissions, they did not produce any results of an actual attack. However, in 2001 at the Cryptographic Hardware and Embedded Systems (CHES) conference in Paris, France, Gandolfi et al. provided some "concrete results" of an actual attack [28]. The attacks were carried out on three different cryptographic implementations. The algorithms attacked were DES, an alleged COMP128 (see [73]) and RSA. The EM radiation was

[19]This does not guarantee success however as will be seen in the next chapter.

picked up with the homemade coil shown in figure 3.10 (taken from [28]) which apparently produced the best results out of a number of antennas tested.

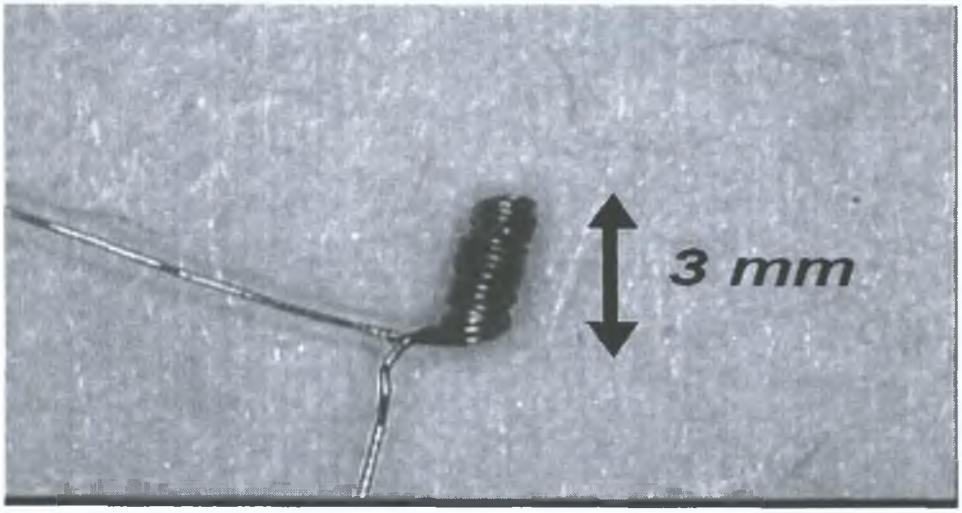


Figure 3.10: Coil used by Gandolfi et al. to carry out EMA attacks on a number of smartcards.

Although it is reported that the key was successfully recovered from each of the implementations, very little information is given about the attack on RSA and it is unclear how the key was obtained in this case. As mentioned in chapter 2 the methods used to obtain the key from a DES implementation and an RSA implementation differ slightly. As well as this, there are a number of methods an attacker can use to break the RSA system.

The main conclusions from Gandolfi et al's paper show that in general EMA is as powerful (if not more so in some cases) than and ordinary power attack. It was pointed out also that a SEMA attack is generally not the same as an SPA attack and that an implementation protected against the latter may not be protected against the former. The localisation properties of EMA was noted as an advantage over normal power analysis. Whereas power analysis shows the combination of all the different signals of the system in one waveform, the electromagnetic sensor can be localised to a position where the signal is strongest thus filtering out any unwanted emissions.

At about the same time, Josyula Rao and Pankagi Rohatgi produced a preliminary report on the leakage of compromising information through EM side channels [74]. Rao and Rohatgi discuss a number of "bad" instructions that leak a lot more information in the EM side channel than through the power fluctuations which may be exploited to circumvent certain countermeasures to power analysis. It is stated in this paper that SEMA attacks can be successful against systems where SPA attacks fail. This confirms the results of Gandolfi et al. in [28].

Rao and Rohatgi used some of the information from the declassified TEMPEST documents found at [50, 52]. The main idea noted from these documents was that the information was not only available at the fundamental frequency of the information being processed but was also available at each of it's harmonics. The harmonics, being higher frequencies radiate with a lot more efficiency than the fundamental and thus pose a greater threat. The signals captured at the different frequencies generally contain different information and by using techniques from multivariate statistical analysis they can be combined in such a way that the information leaked is much greater than any one signal alone.

At CHES 2002 Dakshi Agrawal, Bruce Archambeault and Josyula R. Rao published extended results of Rao's preliminary results [32]. The full report of [74] and [32] can be found at [75]. The ideas presented in this paper are similar to those found in the others however more details of setup are given etc. It is interesting to note that this is the only paper that reports a successful attack on a cryptographic accelerator although no details of the attack are given.

3.4 Summary

This chapter discussed the idea of attacking electronic systems using the electromagnetic radiation which they produce. It initially looked at some background theory on

the causes of the radiation and a number of models were discussed which allow an approximation of the level of radiation emanating from a device. It was seen that there are two main forms of radiation: differential mode and common mode radiation. The former is the easiest to protect against as it can be reduced greatly by proper layout and design. However the latter is more difficult to prevent but is really only a problem at lower frequencies due to it's 20dB/decade roll off above the cutoff frequency.

Next the TEMPEST program was briefly discussed. It was seen that Government organisations have been aware of the compromising radiated emissions for some time and at present are far more advanced than the current state of the art in the public domain. Most of the information found by Government organisations has been fully classified and as a result it is quite difficult to find any useful information.

The state of the art in public domain research was then discussed and although still in it's early stages, it has shown that a problem does exist. The research initially began by studying high voltage devices such as VDUs but was later extended to include ISDN lines and smartcards. Only one source of research was found in which a cryptographic accelerator was attacked. Although it was claimed that this attack was successful, no details were given.

To determine whether or not the DUT was vulnerable to these attacks a number of experiments were setup. The next chapter discusses not only these experiments but also those used for the power attacks described in chapter 2. It will be seen that that although successful in simulated and emulated setups, the attacks against this particular DUT in fact failed to produce the desired results.

Chapter 4

Experimental Setup and Results

The last three chapters gave an introduction and a theoretical grounding for the side channel attacks studied in this thesis. The experimental setup used to carry out these attacks was based around this theory and is outlined in this chapter. The problems encountered will be discussed as will the solutions devised to overcome them. The results of the experiments will be presented and it will be seen that not all were successful. In particular, the attacks against the DUT failed in all instances but were in fact successful in simulations and in a microcontroller design that emulated the operations of the DUT (thus proving that the fundamental techniques were correct). The microprocessor used was similar in structure to those found in most smartcards and therefore provided comparison with experimental results already reported. A number of possible reasons for the failure of the attacks against the DUT are suggested throughout the chapter and suggestions of how an adversary may improve on the techniques are given[1].

4.1 Initial Problems

On beginning this research, little or nothing was known about the techniques or theory involved. The whole area of cryptography was new and it was clear that some basic understanding was needed before an experimental setup could be devised. Initially there was very little equipment available however there was a limited budget that was

[1] It must be noted that failure is considered a positive result in this case as the aim of the experiments was to determine the level of vulnerability of the device, and within the limits of the experimental capabilities, it appeared to be secure.

used for its purchase. Due to this limited budget, the equipment had to be chosen with care - this meant balancing performance with cost. In order to allow an effective balance, it was necessary have a comprehensive theoretical grounding in the relevant areas and to then do a full product review. It is generally the case that theoretical aspects are harder to grasp without having access to equipment to experiment with. This was especially true with this particular research where the element of uncertainty was fairly large. It was very difficult to know exactly what type of filters, amplifiers etc. were needed until some initial experiments were carried out. However, due to the limited budget these experiments could only be setup once a theoretical knowledge was gained so it was a sort of "catch 22" situation.

Unfortunately, a prototype of the DUT was not acquired for quite some time which greatly hindered the progress also and by the time it was received (and its functionality understood), a limited amount of time was left. The cryptographic module itself was a very complex piece of equipment and a full understanding of it was not possible (or necessary) in this situation. When the board was initially received a brief explanation of its operation was provided, however there was still quite a bit of background theory that was required to understand this explanation. Despite this, experiments were begun as soon as the DUT was brought back to the laboratory. The experimental setup was without a doubt the most troublesome and difficult part of this research.

Experiments began by considering the TEMPEST issue. This was chosen first because a relatively greater understanding of communication theory and electromagnetics was possessed than that of cryptography. In hindsight, this was a very naive attitude for two reasons. Firstly, both attacks (power analysis and electromagnetic analysis) required a firm understanding of cryptography and secondly, TEMPEST techniques were nearly all classified and very little information (experimental or theoretical) could be found. On the other hand, power analysis was becoming a very popular area of research and

there was a great deal more information available which would have provided an essential guide in those early stages. By choosing to deal initially with TEMPEST a lot of time was wasted attempting to devise attacks that were in effect already published through power analysis.

4.2 Initial Setup

For the first set of experiments, the equipment consisted of the DUT, an EMC kit, a Tektronix TDS210 oscilloscope and a spectrum analyser. The DUT can be seen in figure 4.1. This board is the original design of this particular model and as can be seen is fairly large and bulky. Although it is not made this way anymore, the operations on the newer versions are much the same.

The board was initially run at a clock speed of 60MHz which caused a few problems as will be seen later. The main system clock is generated within the ASIC (which is under the ASIC fan shown in the diagram) and is not directly accessible to an attacker. As a result, it would be quite difficult to carry out a synchronous sampling DPA attack[2] which would decrease the time for a successful attack.

Unfortunately, throughout the experiment access to the parameters on the board (the key, plaintext and modulus) was quite limited. This was especially true in the early stages of the experiment. Although the board was designed to be controlled by software on a PC this facility wasn't provided. Instead, the board was equipped with a pair of removable flash PROMs that contained control software. These chips are shown in figures 4.1 and 4.2. The normal purpose of the software on these chips was to ensure that the hardware was functioning correctly and was only run during idle periods[3]. It achieved this by calculating the value of $b_t^{e_t} \mod m_t$ where b_t was a test base, e_t a test

[2]This is where the digital oscilloscope uses the DUT's clock to synchronise the data samples of each waveform collected in a power analysis thereby reducing noise. More details can be found in [76].

[3]When the board wasn't being used to encrypt user data.

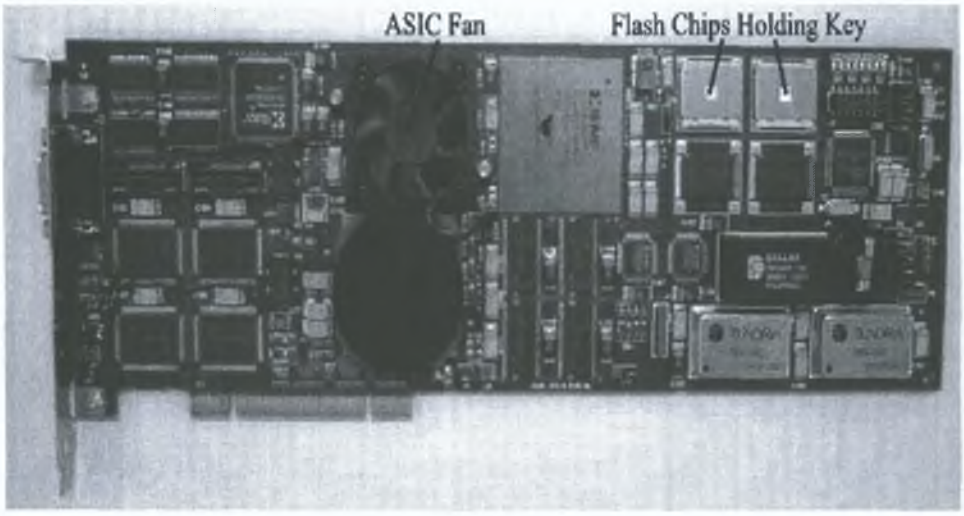


Figure 4.1: Device Under Test (DUT) - This is a prototype Encryption Board but is similar in operation to the marketed versions.

exponent and m_t a test modulus and then testing the result against the actual solution. The actual solution was stored along with b_t, e_t and m_t on the flash PROMs. If the hardware returned a different value than that stored on the chips the processor would notify the user that there was a problem. For this experiment however the PROMS were only used to instruct the board to perform an encryption.

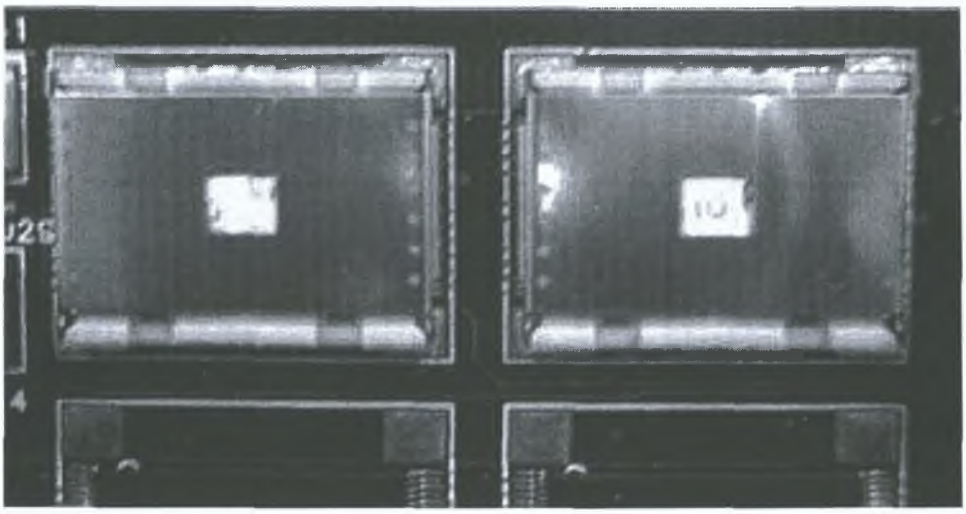


Figure 4.2: The slots on the DUT where the flash chips are seated.

Three sets of flash chips were provided with the board initially, each programmed with a different value of e_t. One value consisted of all ones, a second of all zeros and a third of alternating ones and zeros. An experiment was setup to try to determine if it was possible to tell which exponent was being used. The initial idea was to analyse the spectral content of the radiated emissions from the board whilst operating on each exponent and to determine whether or not any differences existed between them. It wasn't entirely clear how information was going to be extracted from these differences however their existence would have given some indication of the possibility of a successful attack.

A PCI backplane was provided to house and supply power to the board however there was no way of transferring new data. Although not fully desirable, the backplane had three small advantages. It allowed ease of access to the board during the measurements, it mostly eliminated the noise that would have been present from other components inside a normal PC (although its own power supply produced some fairly noisy signals) and it didn't take up any valuable processor time from the PC (which was badly needed for the collection of data at later stages of the experiment). The main disadvantage was that the board could not be controlled by the PC. This reduced the extent to which the experiment could be automated thus increasing the time for an attack. It also prevented testing the board against attacks that required random inputs.

To set up an initial experiment, it was necessary to obtain some sort of antenna, amplifier and a spectrum analyser. Fortunately an EMC kit was available that was designed especially for EMC testing. Its purpose was to allow companies to ensure compliance of newly designed products with the European Union directives (governing radiated emissions) before sending them to an approved testing facility. The probability of failure (and further testing at a later stage) would then be minimised and development costs would be reduced.

The kit is shown in figure 4.3 and consists of five antennas and a battery operated, shielded amplifier. A close up of the antennas can be seen in figure 4.4. Three of the antennas are loop antennas which pick up magnetic fields while blocking electric fields. The other two antennas (the ball probe and the monopole) are used for picking up electric fields while blocking magnetic fields. The size of the antenna determines its localisation properties. The smaller the antenna the more accurately the source of the radiated signal can be found. If the source of the compromising radiated emissions were to be localised to a specific area, then by using the smallest antenna at precisely that area, the time for an EMA attack would be reduced. This is due to the fact that the antenna itself would act as a filter of the surrounding noise whilst passing the signal of interest.

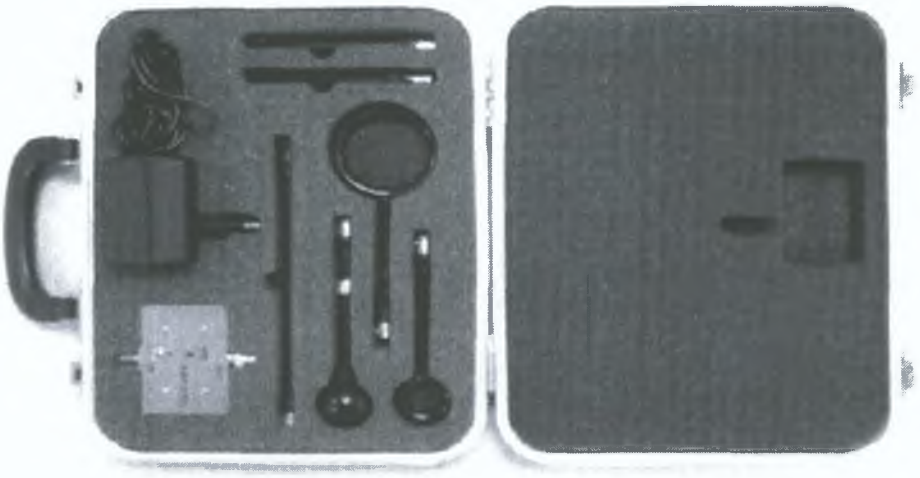


Figure 4.3: The EMC kit used to capture and amplify the radiated emissions emanating from the DUT.

The amplifier is shown in figure 4.5 and as mentioned was battery operated by design. The purpose of this was two fold. Firstly, it eliminated power supply noise and other noise that would normally be picked up by the connecting wires. The second advantage was that it allowed for mobility. Although elimination of the noise was highly

Figure 4.4: Five antennas lined side by side for comparison. The three loops pick up magnetic fields while blocking electric fields, while the ball and the monopole pick up electric fields while blocking magnetic fields. The different sizes allow for more (smaller) or less (larger) precise localisation of the offending emission.

desired, the batteries turned out to be more of a hindrance than a help as they kept running out of power during the acquisition of data (which in most cases took more than 24 hours to complete). Even if the batteries managed to retain their power to some extent, the amplification decreased slightly causing erroneous results. As a very crude solution, two pieces of wire were soldered to the supply terminals for connection to a power supply. This allowed continuous acquisition of data without fear of the batteries running low.

The bandwidth of the amplifier (and the antennas) was approximately 600MHz with its lower cutoff at around 300Hz. It was very desirable that the amplifier block the DC component without blocking any of the lower frequency components because (as will be seen later) these components provided a lot of information about the sequence of events on the board. The gain of the amplifier was 20dB over the pass band which wasn't ideal due the very low amplitudes of the radiated signals. However at that stage no other amplifiers were present. No filters were used for this particular setup as it was

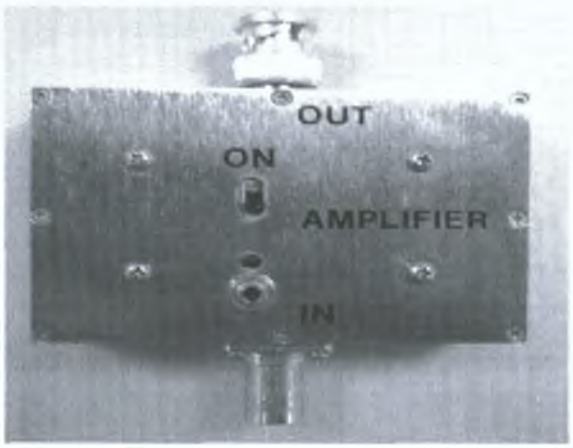


Figure 4.5: 20dB shielded amplifier designed for EMC measurements.

unclear where the signals of interest were located.

The spectrum analyser used was an Anritsu 2661A. This allowed inspection of signals from 9kHz to 3GHz to be viewed with a maximum resolution of 1kHz bandwidth. The emissions at different positions on the board were initially measured directly using the analyser along with the antennas and the amplifier. This was obviously not a very useful method and some way to automate the process was desired.

The spectrum analyser was capable of being controlled remotely by a PC over a serial/parallel connection or a *General Purpose Interface Bus* (GPIB). To this end it was supplied with a "programmers manual" that contained the commands used to control the analyser. This manual was studied initially with an attempt to control the analyser remotely. Some basic commands were sent through the serial connection in order to perform simple actions such as setting the frequency span or resolution bandwidth of the analyser. Although these commands worked as expected, an easier method of accomplishing this was reviewed. After a discussion with a colleague, it was decided that the hitherto unheard of program *Labview* seemed perfect for the job.

Labview is a graphical programming language developed by National Instruments

[77]. It is a fairly user friendly package with all the capabilities needed to control the spectrum analyser and to analyse the data[4]. Programs in Labview are known as *virtual instruments* (vi's) as they are built to resemble actual hardware such as an oscilloscope or spectrum analyser. A powerful capability in Labview is the ability to create "subvi's". These are also vi's however they are used within a larger vi. This allows for a modular design of large and complicated programs. There are two main sections to a vi - the *Front Panel* and the *Block Diagram*. The former is the user interface and acts similar to a real control panel on an oscilloscope or spectrum analyser. The Block Diagram is the inner workings of the program and would be analogous to the electronics within the oscilloscope or spectrum analyser.

After learning the basics of Labview, a program was built to control the spectrum analyser. Initially it was attempted to use Labview to send the commands to the spectrum analyser and a way was found to do this. However, a far easier solution was simply to download the Labview drivers for the analyser from the National Instruments website [77]. These were simply pre-packaged subvi's containing the commands for controlling the analyser. From the users point of view, they were a series of "black boxes" which accepted the required configuration data (such as resolution bandwidth etc.).

The Front Panel of the particular vi used for controlling the spectrum analyser was rather bare compared to most due to the fact that the initial parameters set had no need to be changed in future runs of the program. As a result it is not shown here but a different example of a Front Panel and its corresponding Block Diagram are given in section 4.3.2 (figures 4.19 and 4.18).

The final experimental setup for this particular part of the research is shown in figure 4.6. Each antenna was connected to the EMC amplifier using a 50 Ω coaxial cable.

[4]However at the time a greater knowledge of the mathematical package Matlab was possessed and the data analysis was carried out using this.

As the impedance of both the antenna and the amplifier were both 50 Ω, this allowed for maximum power transfer between the antenna and the amplifier. The output of the amplifier was also rated at 50 Ω and was connected to the 50 Ω impedance of the spectrum analyser through a coaxial cable so the complete system was matched. Although no filters were used at this stage the amplifier and antenna setup produced a bandlimited signal, however it was doubtful that much information was lost as a result.

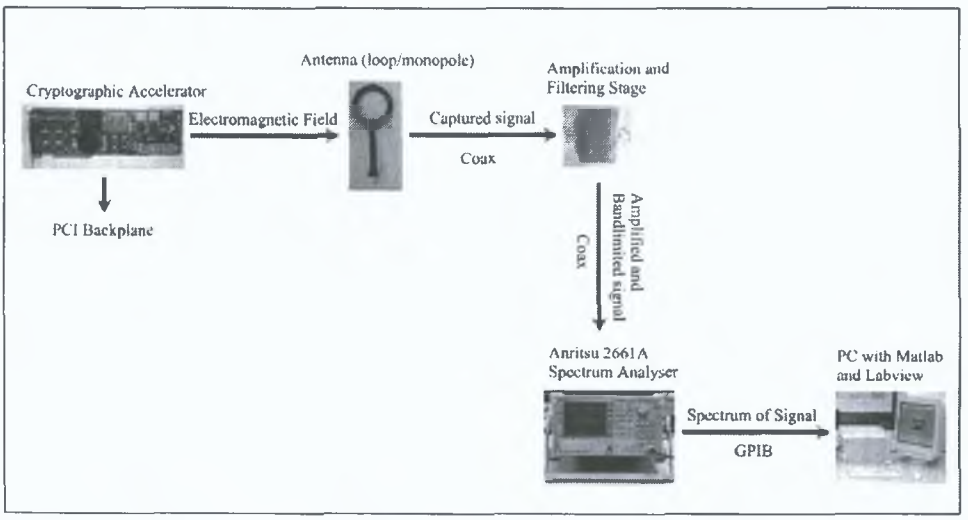


Figure 4.6: The initial experimental setup. The antenna was used to pick the magnetic (loop) or electric (monopole/ball probe) field energy surrounding the board. This was then amplified and passed into the spectrum analyser. The analysers measurements were then sent over GPIB to a PC for analysis with MATLAB.

As shown, it was decided to connect the analyser to the PC using the IEEE-488 standard (also known as GPIB). This allowed for much faster transfer rates than either the serial or parallel connections. The GPIB standard was originally developed by Hewlett-Packard in the 1960's and was known as HPIB. Using GPIB, the PC can control up to 15 instruments and the bus cable can be up to 20 meters long [78]. However for this particular setup it was only required that one instrument be controlled. The GPIB commands were contained within the drivers downloaded from the National Instruments website which was very convenient.

Under the control of Labview, the spectrum analyser was instructed to sweep a portion of its bandwidth and send the results to the PC for analysis. The total frequency range analysed was from DC to 1GHz. Although the spectrum analyser was rated from 9kHz to 3GHz, the DC value was used for convenience and the maximum value of 1GHz was used simply because the antennas and amplifier had a bandwidth of only 600MHz (although this was the 3dB bandwidth, components could still be identified far beyond this frequency however it was understood that these were not correct measurements). Setting the frequency span of the spectrum analyser to 1GHz (beginning at DC and ending at 1GHz) would have been useless as in order to set a realistic sweep time[5], the resolution bandwidth had to be set to about 10Mhz. This meant that only components greater than 10MHz apart could be separately identified. To overcome this, the Labview program was setup in a loop so that a reasonable span was swept in each loop. On each subsequent loop, the end frequency used for the previous span was set as the start frequency for the next. This allowed resolutions of up to 1kHz to be used if desired. It was not necessary to have this high resolution and for the first few loops it was set to 3kHz. The resolution and the frequency span were then increased in steps as the frequency was increased.

The program was run a couple of hundred times for each key and each set of results averaged. The spectra captured whilst processing the key's with all zeros and alternating ones and zeros were then subtracted from that captured whilst processing the key consisting of all ones. The idea was that any spectral content present in one key that wasn't present in the other keys would manifest itself in the difference trace. At the time this experiment was carried out, the idea of power analysis had not been studied. It was only realised later that a similar idea was used in the time domain for power analysis.

[5]The time it takes for the analyser to sweep from the start frequency to the stop frequency.

Although the idea sounds simple enough, due to extensive frequency drifts, the difference spectrum contained values that were not zero when they should have been. This greatly affected the programs operation. It was decided then to plot the spectra on the same graph and to analyse the results manually to try to identify any differences in the spectral content. However, although a great deal of time was put into this, no differences were found and all three signatures appeared to be the exact same. It is very likely that this wasn't really the case, however due to the limits of the equipment available no differences were observed.

As an example, figure 4.7 shows the fundamental clock frequency captured using all three keys. It can be seen here that the signal contains sidebands which may or may not be data modulated onto it and may in fact only be phase noise. A close up of the graph revealed the frequency content to be the exact same for each signal and this didn't reveal much information due the missing phase information.

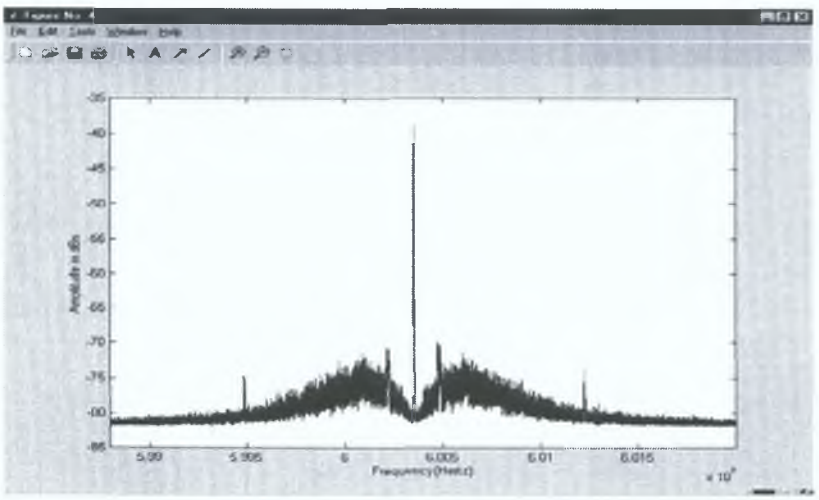


Figure 4.7: This is the fundamental frequency of the 60MHz clock. There appears to be sidebands on either side of it which may have contained information.

Figures 4.8 and 4.9 show the second and third harmonic of the clock. It can be seen that the second harmonic is actually larger than the third which is not normally expected for

a square wave and only odd harmonics should exist. One possible reason for this can be seen from equation 3.5 in chapter 3 where the duty cycle of the signal determines the level of the harmonics. If the duty cycle is not exactly 50% then the signal may contain even harmonics which may be larger than the odd harmonics. This can be seen in figure 3.4.

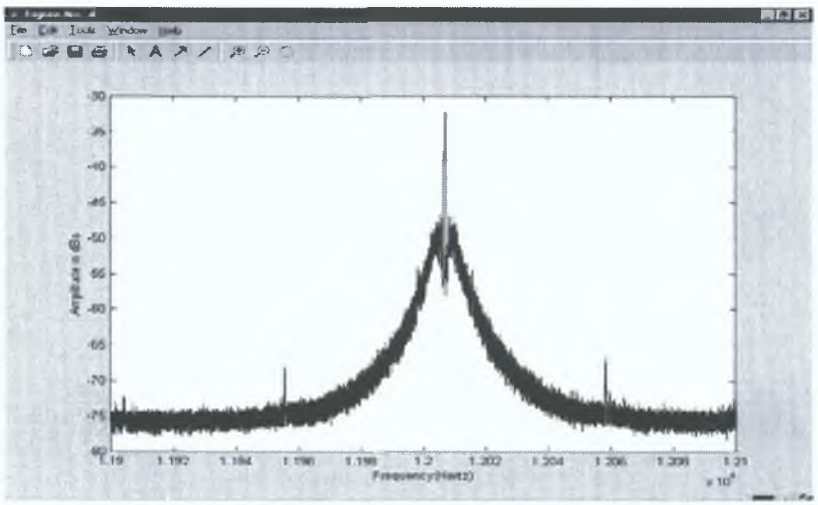


Figure 4.8: The second harmonic of the 60MHz clock. This is actually larger than the third harmonic which suggested that the clock did not have a 50% duty cycle.

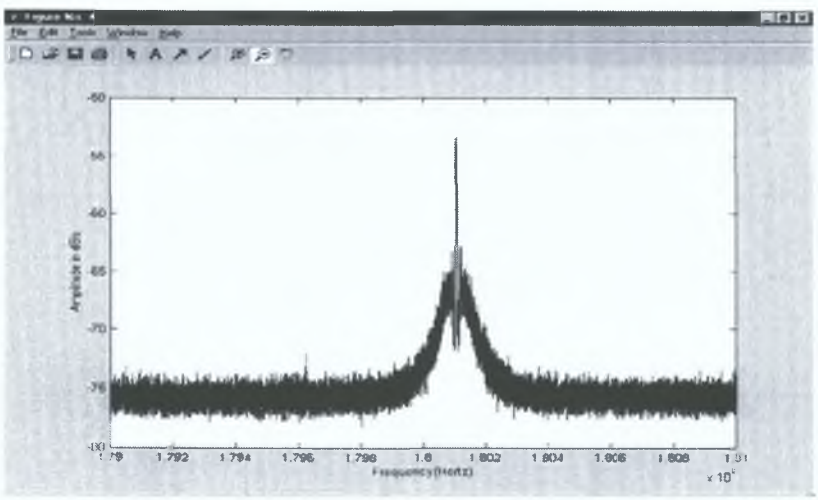


Figure 4.9: The third harmonic of the 60MHz clock.

Signals with sidebands were of a particular interest as these may have been modulated with the key data. Of particular concern were harmonics of the 60MHz clock frequency. Although these were noticed, it was not clear whether or not these sidebands contained useful information. As will be seen an experiment was setup to determine this however no useful results were obtained.

4.3 Power analysis

Having failed the initial electromagnetic analysis it was decided to try to analyse the power being consumed by the board. A number of problems needed to be resolved before this could be accomplished. The first problem was that of accessing the power signal of the board. As mentioned, the DUT was connected to a PCI backplane with no easy way of connecting a small resistor in series with the ground connection (as is generally the procedure when attacking smartcards). At first it was considered building some form of extension that would allow a resistor to be placed in series with the ground line, however a commercial product was found that allowed measurement of this power without any resistor. The device can be seen in figure 4.10 and is known as a *universal extender board*. This board sits between the PCI backplane and the DUT as shown in figure 4.11 (this picture also shows the PCI backplane used to operate the DUT).

The extender board was purchased from Ultraview Corporation [79]. It is generally used for testing newly designed PCI boards as it allows (amongst other things) power consumption limits to be placed on the board under test thereby saving irreparable damage should a fault occur. It is referred to as a universal board simply because it caters for both 3.3V and 5V PCI environments as well as catering for 32 and 64 bit boards. It was purchased for this project as it also allows ease of measurement of the power consumed by DUT.

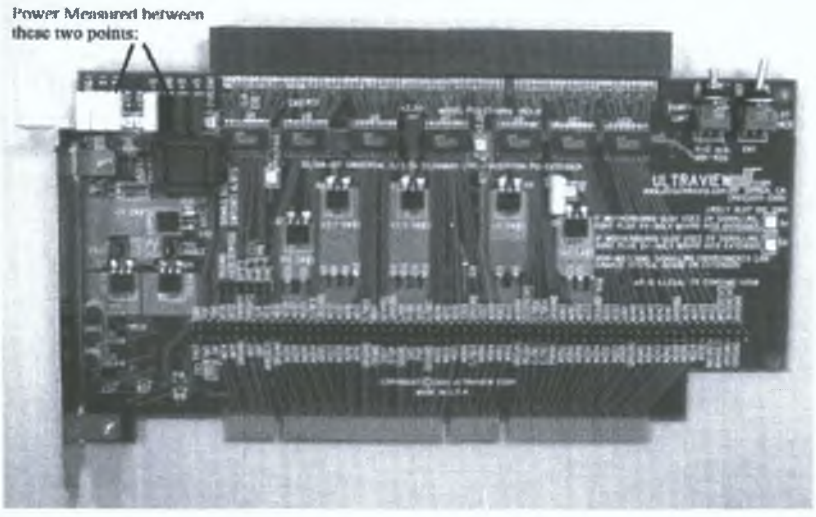


Figure 4.10: Extender Board used to measure the power consumption. The DUT sits on top of this. The third slot in the top left hand corner allows observation of the 3.3V power consumption (which is of concern here) with respect to ground (the slot two to the right of it).

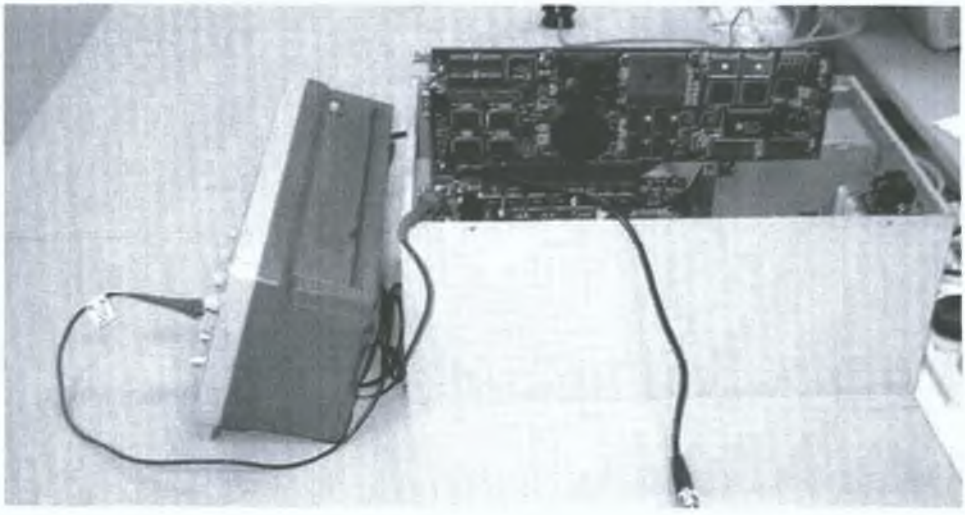


Figure 4.11: Side view of the PCI backplane holding the extender board and the cryptographic module.

With the problem of accessing the power fluctuations solved, the next problem was the measurement of these fluctuations. The oscilloscope that was initially available was a Tektronix TDS210. This had a bandwidth of 60MHz and a *maximum* sample rate of $1GSs^{-1}$. The DUT was run while inserted in the extender board and the power

drawn from the supply measured. Initially nothing was observed but a large DC signal, however once AC coupling on the scope was enforced, the power signal became clear.

It was desired (as before) to transfer the information to the PC for analysis. A program called *Wavestar* was supplied with the oscilloscope. This allowed the displayed trace to be transferred to a PC over a serial link or GPIB and was initially used to save the effort of having to write a program in Labview to do the same thing.

It became clear almost straight away that the oscilloscope and the software were not good enough for the job at hand. Although the waveform was transferred to the PC, it was quite awkward to capture the actual data values into a file for processing by Matlab or Labview. It could be copied and pasted without a problem but this was of no use for an automated setup. Another problem was the fact that the data only contained 2,500 points. This was an extremely low number of points and was absolutely no use in a power attack[6]. The sample rate of the scope was determined by these data points using the following equation

$$f_s = \frac{2500}{10 \times t_b} \tag{4.1}$$

where t_b is the timebase of the oscilloscope measured in seconds/division. The 10 is derived from the fact that there are 10 divisions on the oscilloscope. As a result there was very little control over the sample rate and none over the amount of points that were taken.

Using equation 4.1 it might be considered that the minimum time base setting would have been 250ns per division however the oscilloscope used an interpolation algorithm and could actually reduce the value of t_b to 5ns per division however the sample rate is still only 1GSs^{-1}. Aliasing is avoided due to the fact that the analog bandwidth is only 60MHz which is well below the theoretical maximum value of 500MHz (half the maximum sampling rate).

[6]In fact, as will be seen later, the number of points captured in the final setup was 3,000,000!

Even though a better oscilloscope was needed, a small amount of information could be gained from the TDS210 by analysing the waveform displayed however a proper attack would never have been achievable. As well as the extremely poor 2,500 acquisition points its 60MHz bandwidth would have filtered out most of the harmonics of the DUT as its clock speed was also 60MHz as mentioned.

Although there were higher bandwidth oscilloscopes available, there was never the right combination of features that would allow for a power analysis. It was decided that a new oscilloscope should be purchased. An initial search showed that the prices of the scopes with the right specifications were beyond the budget that was available. Even at the these prices the specifications weren't ideal. After some more searching, a company called *Gage Applied Inc.* was found that produced hardware digitisers [80]. These are basically oscilloscopes that fit into the PCI slot of a PC. The specifications were excellent and almost perfect for this experiment, however it was still beyond the budget available. One of the boards real time sample rates was 2GSs^{-1} and had an external clock input which would have allowed for a synchronous attack. Both of these specifications would have reduced the time for an attack immensely.

The main cost factor in high speed digitisers is the speed of the memory. If a board samples at 2GSs^{-1} then the memory has to have an access time of at least $1/2\text{GSs}^{-1} = 500\text{ps}$. Memory of this speed is extremely expensive and adds heavily to the cost of the digitiser (or oscilloscope). Luckily however, a way was found to sample at 2GSs^{-1} without needing such high speed memory. After a review of some of the other products on the market, it was discovered that National Instruments also produced high speed digitisers. Unfortunately the specifications weren't as advanced as some of those from Gage Applied Inc., however a digitiser was found that was capable of sampling at an *effective* maximum sample rate of 2.5GSs^{-1} even though its Analog to Digital Converter (ADC) was only capable of sampling at a maximum rate of 100MSs^{-1}.

This was achieved using a technique known as *Random Interleaved Sampling* (RIS) which is a form of *Equivalent Time Sampling* (ETS) (see [81] for more information on ETS).

RIS allows for sampling rates of up to 25 times the maximum sampling rate of the ADC. Of course this performance comes at a cost - the signal being sampled must be periodic. Initially it was considered that this would be useless as generally the key data being attacked was not periodic. However, if the same values are calculated over and over then the waveform itself would be (in effect) periodic and RIS would work. The sample rates allowed using RIS were integer multiples of the maximum *real time sample rate*[7]. These integer values were known as *oversampling factors*.

The specific digitiser used was the NI5112 and is shown in figures 4.12 and 4.14 along with a block diagram in figure 4.13. At the time of purchase, this was the highest quality high speed digitiser on offer by National Instruments. It had a resolution of eight bits which wasn't ideal however to increase this value and keep all other aspects equal would have increased costs significantly.

One of the advantages of the NI5112 was that it had 16MB of onboard memory. This allowed a *record length* of 16 million samples and can be compared to the 2,500 offered by the TDS210. As the waveforms collected from the digitiser were sampled at very high rates, it was necessary to have such deep onboard memory if any reasonable time slice was to be obtained. The digitisers vertical input range was adjustable from $\pm$25mV to 25V. This allowed voltages as small as $50\text{mV}/2^8 \approx 200\mu\text{V}$ to be identified. According to Messerges et al. the average amplitude of the bias spikes in the differential trace (for the particular setup reported in [24]) was 6.5mV. Although it was highly unlikely that the same value would be observed for the DUT, it was assumed that it wouldn't have been much different. As a result, eight bit resolution should have been

[7]The real time sample rate was that of the ADC.

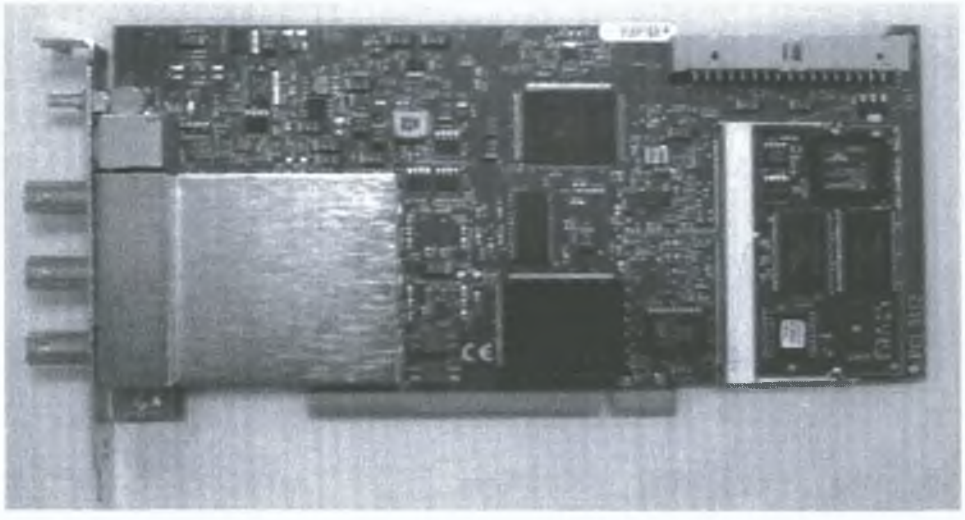


Figure 4.12: Close up of NI5112 digitiser. The three BNC connectors are channels 0, 1 and an external trigger. There is also and SMA connector.

sufficient (especially with some amplification).

The fact that the card was based on the PCI bus architecture was initially considered an advantage as it would have allowed much faster data transfer rates than the GPIB connection used previously. However, as will be seen later, the greatest hindrance to speed in this whole setup was the RIS mode and in some cases required up to four days to collect one portion of data.

4.3.1 Initial SPA Setup

Having obtained the extender board and the NI5112 digitiser, a power analysis experiment was setup. SPA was the obvious choice for a first attack and even though it was highly unlikely to reveal any secret information, some noticeable correlation between the power waveform and the theory of operation was expected. The information gained would then assist in the experimental setup for a DPA attack.

Figure 4.15 shows the setup used for the attack. It can be seen that the DUT slots into the universal extender board which allows measurement of the power consumption.

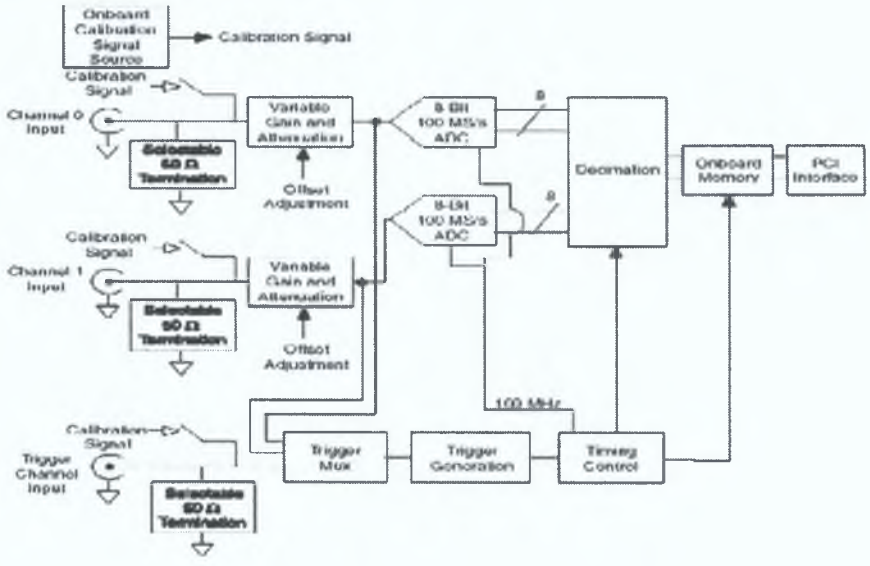


Figure 4.13: Block diagram of inner workings of the NI5112. As can be seen there are two input channels as well as an external trigger channel. The input impedances are switchable between 50Ω and 1MΩ.

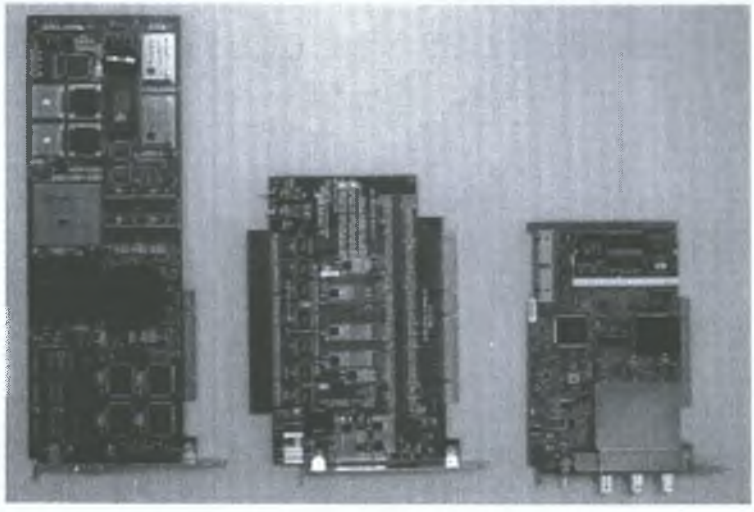


Figure 4.14: All three PCI boards placed side by side for comparison. From left to right: DUT, Extender Board and NI5112 digitiser.

The measured signal is then passed to an amplifier and filter stage which is then passed to Labview for display. No mathematical analysis was performed (or required) at this stage as the appropriate software hadn't been written.

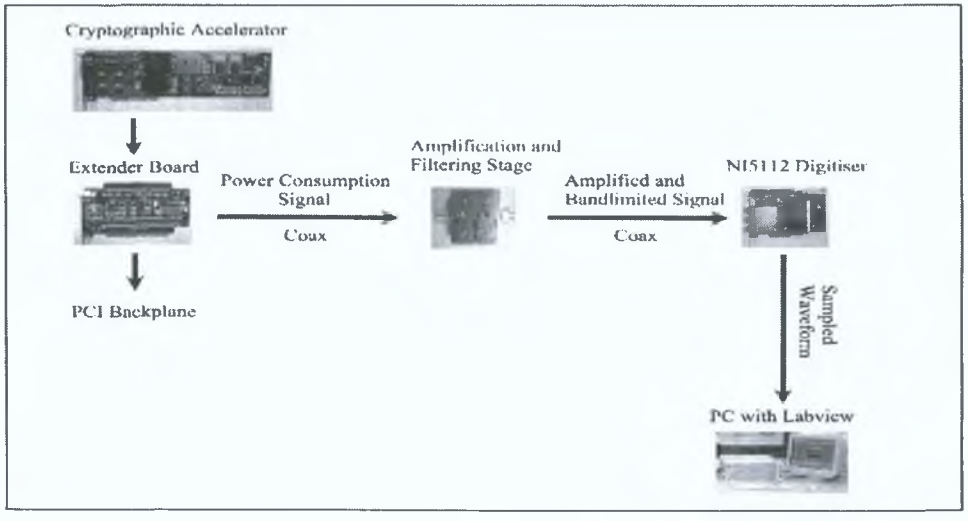


Figure 4.15: Block Diagram of DPA setup.

As mentioned in section 2.5 the board implemented a version of Montgomery's algorithm for modular exponentiation. This basically meant that it calculated the value of $b^e \bmod m$ without the need for division. The particular version of the algorithm used was the left to right square and multiply and therefore the hardware operated on each bit individually, starting with the LSB. The multiplication operation (see table 2.7) was performed for every bit regardless of its value[8]. If the value of the bit was a 1 a register update occurred, however this did not take up an extra clock cycle as might be expected. Due to this fact, it is unlikely that a timing attack (see [82]) would have been successful against this board as equal lengths of time were required to operate on a 1 and a 0. The performance of the multiplication on every round and the fact that an extra clock cycle was not needed for the register update were both part of the possible reasons that the attacks were unsuccessful.

[8]In smartcards this generally would not have been the case due to a lack of resources.

The multiplication operation took exactly

$$\begin{aligned} N_c &= 2 \times (N_b + 2) + 1 \\ &= 2 \times (2174 + 2) + 1 = 4353 \end{aligned} \tag{4.2}$$

clock cycles, where N_c is the number of clock cycles and N_b is the number of bits in base (which was 2,174 in this case). The board was running at a speed of 60MHz so this meant that the length of time for one multiply was

$$\frac{4353}{60 \times 10^6} = 72.55\mu s.$$

The number of bits in the exponent was $N_e = 2,176$ so the total time for one complete exponentiation was

$$72.55 \times 2176 \approx 158\text{ms}$$

Having calculated this result, it was hoped that the power waveform would show some relationship to it. As mentioned, on the initial measurement any fluctuations were hidden amongst the relatively large DC component. Once this component was filtered out the waveform of figure 4.16 was observed. From the figure it can be seen that the width of the pulse was exactly as calculated. This proved beyond doubt that this waveform represented one complete encryption operation. Knowing the shape of the pulse and its duration, it was possible then to carry out an DPA attack, as the rising edge of the pulse could be used as a trigger.

The pulse of figure 4.16 was captured using a sample rate of 1MSs^{-1} and exactly 250,000 sample points were collected[9]. This gave a time period of 250ms. If this same waveform had been captured with the maximum real time sampling rate of the digitiser

[9]It was realised that aliasing would occur with this setup however the method was just to show that the pulse length was as calculated.

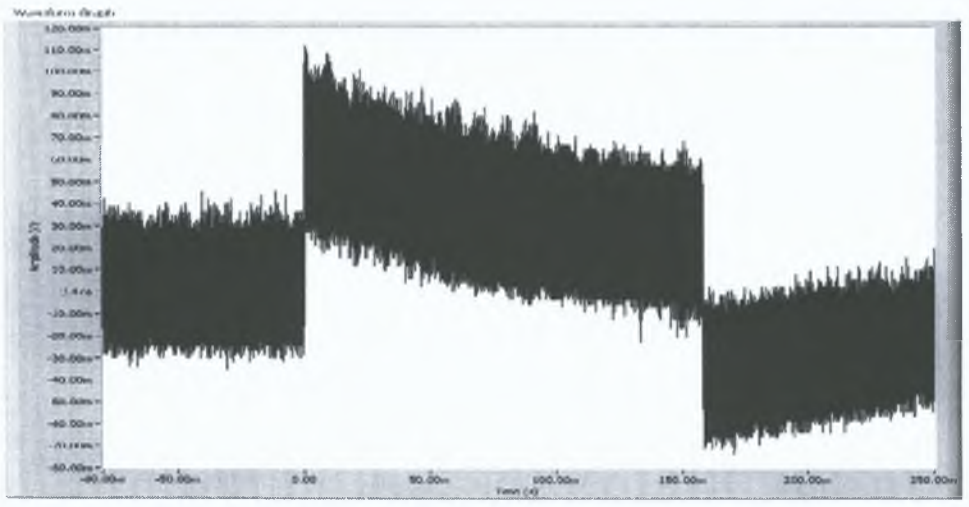


Figure 4.16: This is a 158ms pulse in which one complete encryption is performed. The amount of memory needed to capture this entire waveform at 100MSs^{-1} would have been impractical, so only sections of it were sampled for the attack.

(100MSs^{-1}), the number of points required would have been 25 million! The onboard memory of the digitiser was 16MB so it would have been impossible to collect this number of sample points. As a result it was realised that to carry out a DPA attack some sort of "sliding window" method would need to be used.

It was interesting to note the time between encryptions and to see this a much greater number of data points were collected. Figure 4.17 shows the results, where it can be seen that the time between the start of one encryption and the start of the next is 812ms. This is an extremely long length of time for a cryptographic accelerator to have to wait between encryptions. However, as these particular operations were only run as tests, this didn't affect the operation of the board.

Another waveform was then collected at 100MSs^{-1}. This time the number of points was greatly reduced. Each operation on the exponent bit was visible to an extent, revealing information about the length of time it takes to perform. The time between exponent operations was seen to be $72.55\mu s$ as calculated above.

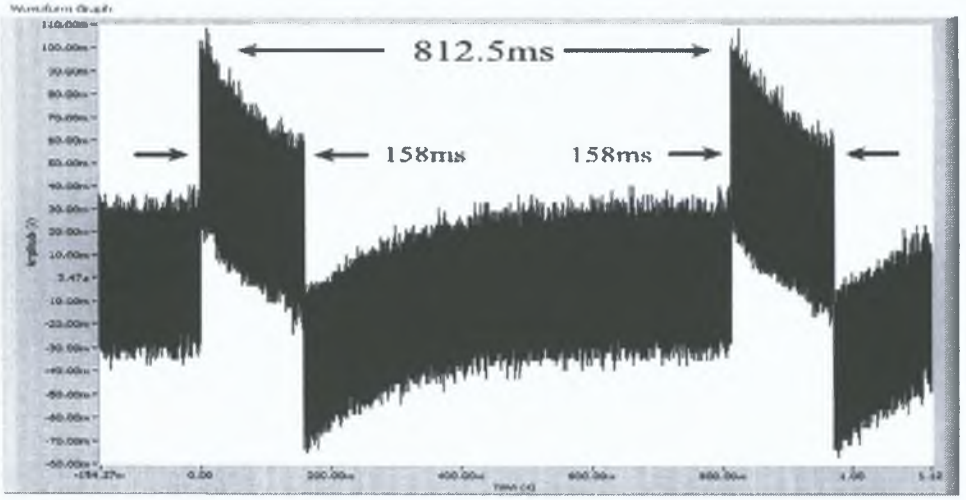


Figure 4.17: Two encryption pulses showing the time difference between encryptions.

Although it was attempted to use RIS to capture some of the waveform, the pulse edge wasn't steady enough to allow a this to work. As a result the maximum sampling rate available was $100MSs^{-1}$. This was a problem as aliasing would occur if the signal contained frequency components over 50MHz. To prevent this happening a 45MHz bandpass filter was used to block any signals above this frequency. It was realised that there was a good chance that a lot of information would be filtered out however it was carried out for completeness.

4.3.2 Simulated setup

Before attempting a DPA attack, a new Labview program had to be written that would implement the analysis of the data. Once the program was written it was necessary to test its functionality. To this end, another Labview program was written to simulate the waveforms produced by the DUT. It wasn't necessary for the simulated waveforms to exactly model those produced by the DUT (which would have been extremely difficult), however they did have to have a number of things in common as will be seen.

As discussed on a number of occasions, a power attack generally involves sending a

large number of plaintext (or ciphertext) inputs to the cryptosystem being attacked and analysing the resulting power fluctuations. It was therefore necessary that the simulation produced waveforms containing the same key information every time, however the noise had to be random.

The waveform was not only to include sources of noise such as external and thermal noise, but it was also to include *algorithmic noise*[10]. This was noise caused by the manipulations of the data received (i.e. the message to be encrypted) and would be different for different values of the plaintext. If the same plaintext were received on each run of the algorithm this noise would be not be reduced by averaging.

Due to the register update of the function exp_always_mul() in figure 2.7, theoretically there should have been a bias spike every time the exponent bit was 1. For the key containing all ones, pulses should have occurred every $72.55\mu s$ whereas for the key containing alternating ones and zeros it should have occurred every $2 \times 72.55\mu s = 145.10\mu s$. In the last key, containing all zeros, no power spikes should have been observed. This was all taken into account in the simulation.

Figure 4.18 shows the Front Panel of the main program for producing and saving the simulated data. As can be seen, it had a number of controls and indicators as any normal physical instrument would. There were no graphs built onto this Front Panel as they were not needed in this case. All that was required here was for this program to produce a user selectable number of waveforms for one of the three keys. The simulated waves, once produced, were then saved to three separate files.

Each of the signals sought were produced without any noise (which was added later). For the exponent containing all ones, a pulse was produced every $72.55\mu s$. The width

[10]Messerges describes five types of noise that can occur in power waveforms: external, intrinsic, quantisation, sampling and algorithmic [76]. Sampling and quantisation noise were not included in the simulation for simplicity, however this should not have affected the programs operation provided it was taken into account in the actual setup.

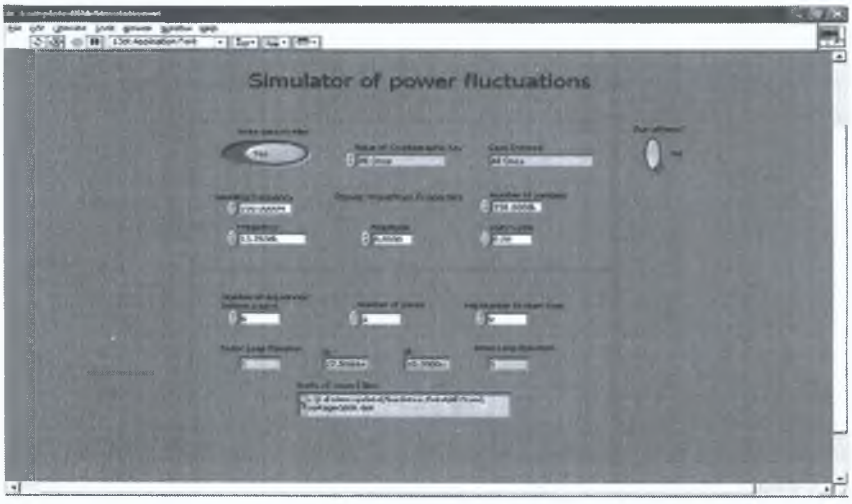


Figure 4.18: The Front Panel of the main vi for simulating the power waveforms produced by the DUT.

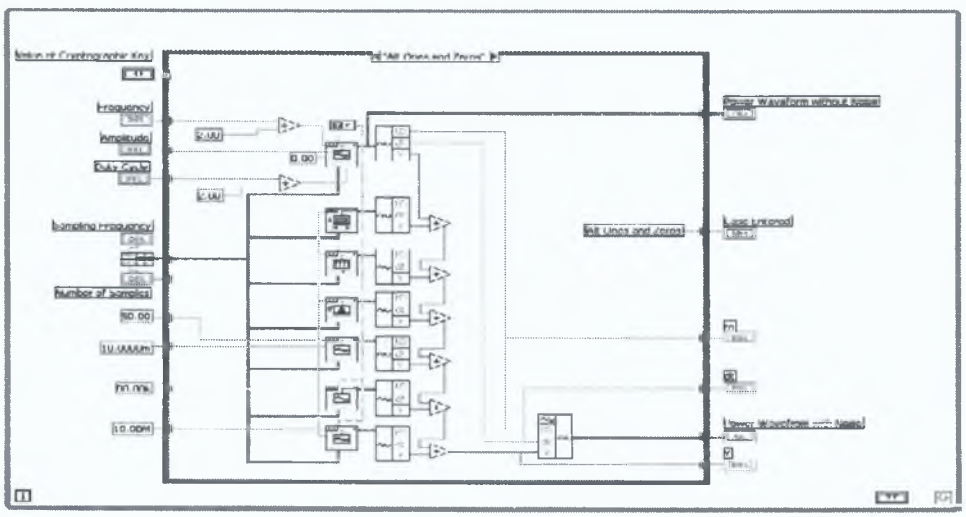


Figure 4.19: This is one part of the Block Diagram of Labview program for producing the simulated power waveform.

of this pulse was

$$\frac{1}{f_c} = \frac{1}{60 \times 10^6} \approx 16.7\text{ns}$$

where f_c is the clock frequency. It was not known whether or not this pulse width was accurate but it was assumed to be approximately this value due to the register update

lasting only one clock cycle. The exponent consisting of alternating ones and zeros contained a pulse every 145.10μs as discussed above and that consisting of all zeros contained no bias spikes. The pulse widths were the same for each case.

The simulated waveform for the exponent containing all ones is shown in figures 4.20 and 4.21. The first graph is simply the power spikes produced without any noise present ("noise" being any signal other than the spikes themselves). The exact shape of the real spikes was unknown and the likelihood was that they were not flat pulses. They would generally be different for different cryptographic modules and would be determined by such things as internal inductances and capacitances (both parasitic and designed). However, the pulse shape should not have affected the simulations purpose. Figure 4.21 shows the same signal except that this time a lot of noise has been added. The noise consisted of a number of random sources as well as a number of periodic sources. The periodic sources simulated both internal and external periodic sources that might have been present in a real attack. The main difference being that most of the internal periodic signals will have some sort of phase relationship with the spikes of interest whereas the external periodic signals will not (an example being the 50Hz pickup from the power supply). The internal sources would therefore not have been reduced by averaging.

The number of simulated waveforms produced in each exponent was one thousand. As there were three exponents, three thousand files were produced for on each run of the program. The results of averaging over the thousand simulated waveforms for the exponent consisting of all ones can be seen in 4.22. Comparing this to figure 4.21 it can be seen that the noise has been greatly reduced, however it still masks the bias signal due to the bias signals very low level.

Autocorrelation was also used to try to pick the signal of interest out of the noise. However, the other periodic signals present within the noise were picked up instead.

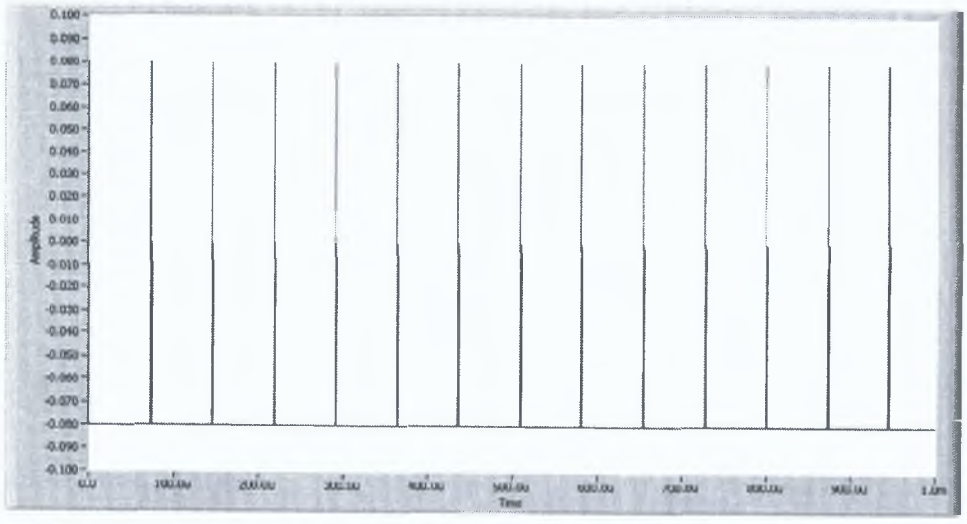


Figure 4.20: The signal here is the simulated version of what might have been produced (without any noise) using a key with all ones. The spikes had an amplitude of 160mV and were offset by -80mV. They lasted for only 16ns which was a mark-space ratio of only .01 percent (as the period was 72.55μs).

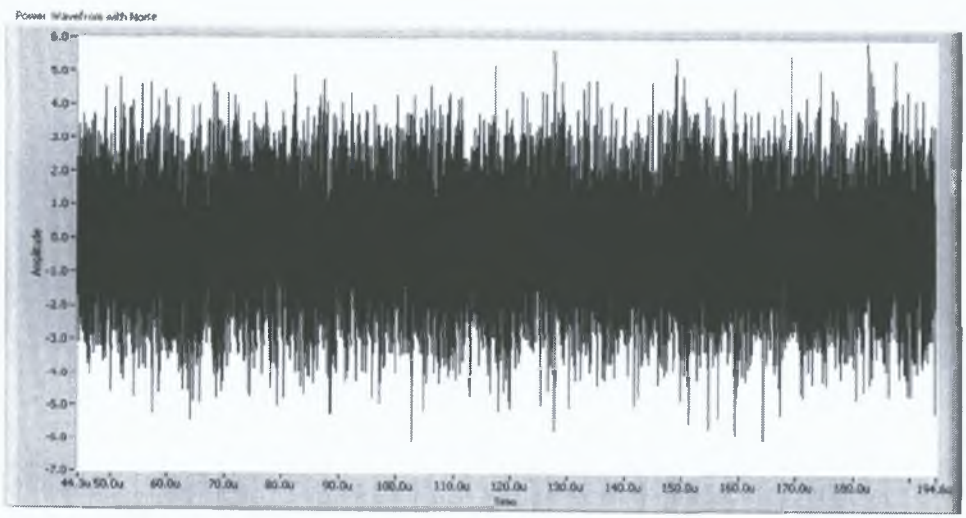


Figure 4.21: This graph shows the one in figure 4.20 with an excessive amount of noise added. This signal can be seen to exceed $\pm$5V at some points and completely masks the power signal sought after.

This is where the differential trace (equation 2.12) helps tremendously as it will get greatly reduce any periodic signals not correlated with the key.

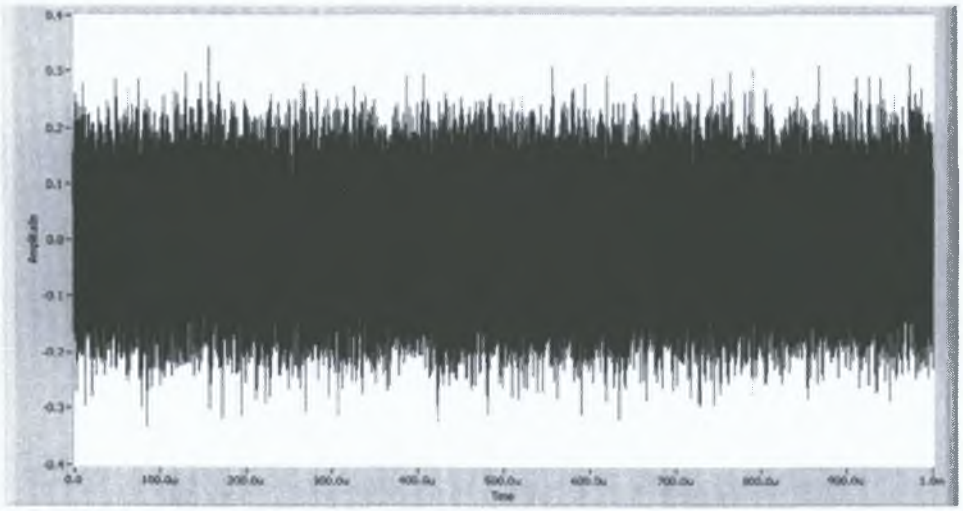


Figure 4.22: This graph shows the quality of the signal produced using the "All Ones" simulated data after 1000 averages. The noise has been reduced significantly however it still completely masks the power signal sought after.

After 15,000 averages, the waveform shown in figure 4.23 was produced. There was an 80kHz signal that was in synchronisation with the operations of the board. As a result the averaging did not reduce this waveform and it still hid the bias spikes. On the other hand the sources of random noise were reduced to an insignificant level and all that remained was to remove the unwanted periodic signal. To accomplish this it was necessary to obtain a differential trace. This was done by averaging 15,000 simulated waveforms using the key consisting of all zeros and subtracting this from that shown in figure 4.23. The resulting trace can be seen in figure 4.24 where the key data is now clearly visible[11].

It was realised that the simulated setup was an over simplified version of the DPA attack and that it would be virtually impossible to tell the key by simply looking at the position of the power spikes. However, the simulation showed that software written to carry out a DPA attack was functioning correctly and therefore fulfilled its purpose.

[11] Similar results were also obtained using the key consisting alternating ones and zeros however they have been omitted for simplicity.

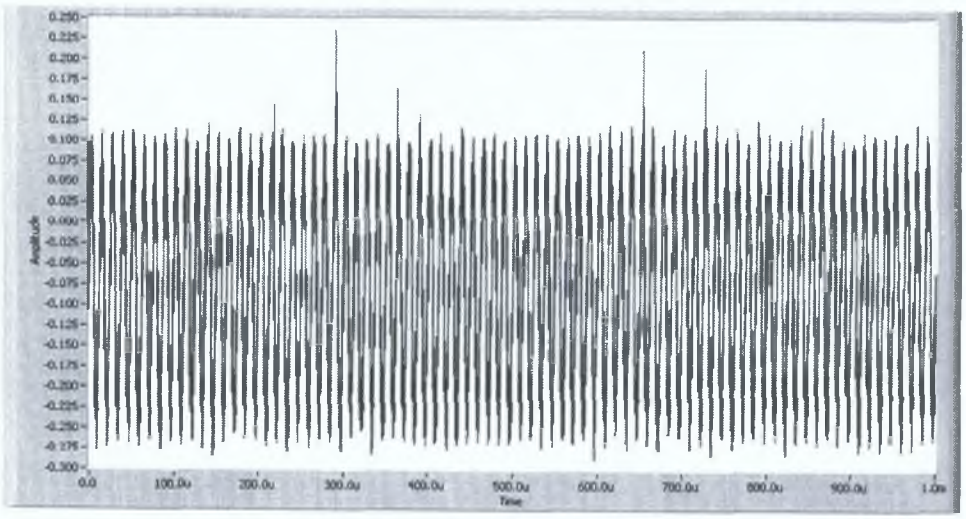


Figure 4.23: This graph shows the quality of the signal produced using the "All Ones" simulated data after 15000 averages. The random noise has been reduced to insignificant levels however a large 80kHz wave can be seen that masks the bias spikes.

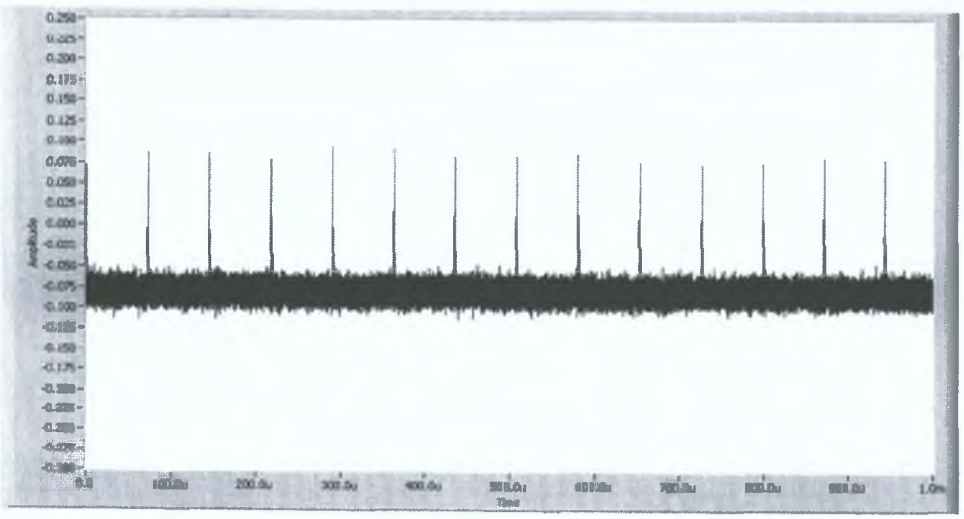


Figure 4.24: This is the differential trace produced by subtracting the average of the waveform produced by the key with all zeros from that with all ones. The key is clearly visible thus verifying the integrity of the Labview program.

The next step was to attempt an actual attack.

4.3.3 Initial DPA attack

Having tested and verified the programs functionality, it was time to attempt a real DPA attack. Three main problems still remained. The first was the fact that only changes that could be made to the parameters of the system were through three sets of flash PROMs. Increased access had been sought at this stage however it had not been received due to a number of delays. The second problem was the fact that the RIS mode of the digitiser required a steady trigger. This meant that the maximum sample rate was $100MSs^{-1}$ which was clearly not enough for a signal that would contain harmonics well past 50MHz. The third problem was the fact that the board was run at 60MHz and that the analog bandwidth was only 100MHz. If there were harmonics of the clock signal that contained information these would be filtered out.

Despite these problems a power attack was attempted. The experiment was setup as in figure 4.15 and the Labview program run 10,000 times for each exponent. The falling edge of the 158ms pulse shown in figure 4.16 was initially used as a trigger. It was important therefore that this was not filtered out. At one stage, two amplifiers were used in succession to try to improve the results however, the lower cutoff frequency of the second amplifier was about 100kHz which completely filtered out the rising and falling edges leaving nothing to trigger off. One of the solutions attempted was to take another probe from the same 3.3V output and place it into the external trigger section of the oscilloscope. This should have allowed triggering however the two probes in parallel loaded the signal too much and it ended up being more of a hindrance. As a result the only amplifier that was available to use at this stage was the EMC amp because its lower cut-off frequency was 300Hz.

The attack was unsuccessful which was expected due to the low sampling rate, low analog bandwidth of the digitiser and other problems discussed so far. Figures 4.25 shows the average of the waveforms collected using the key consisting of all ones. The

intrinsic and external noise had been reduced to insignificant levels and it was possible to identify the exponent operation which occurred every $72.55\mu s$. There was a 90kHz signal that was not reduced by the averaging suggesting that it was dependent on the operations of the board. However, as it had nothing to do with the information sought it was considered as noise. A similar situation arose in the simulation and was remedied by obtaining the differential trace however the differential trace in this case consisted of just noise and no information could be found.

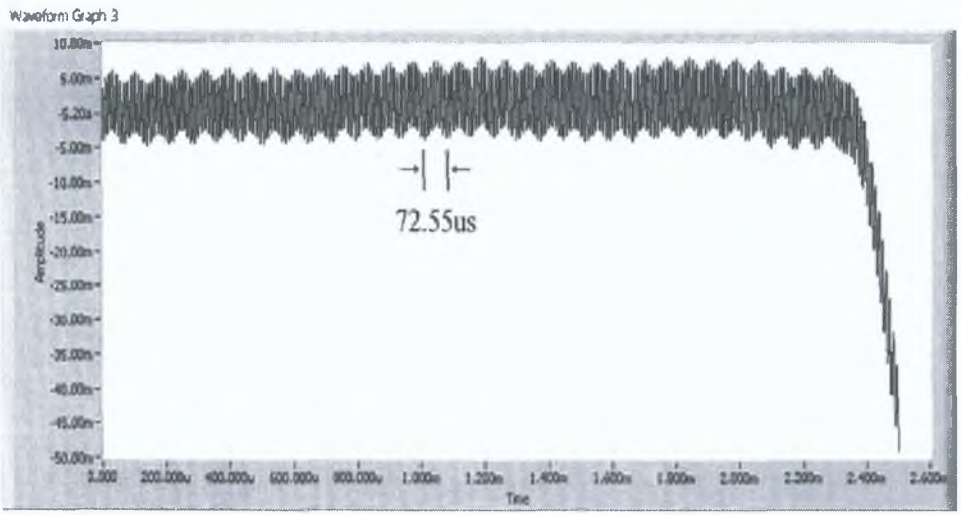


Figure 4.25: Average of the waveforms produced by the exponent with all ones. There is a signal of approximately 90kHz effecting the reading. Each exponent operation can be barely distinguished as the envelope of the 90kHz signal.

One of the main problems was the very large trailing edge seen in figure 4.25. This meant that the vertical range had to be set larger than was necessary. As a result the quantisation noise was increased and it was very possible that the bias spikes would be lost altogether in the quantisation process. To remedy this, the results were then taken at the center of the 158ms pulse, far from the rising and falling edge. This was accomplished by triggering off the rising edge of the pulse and using the trigger delay option to capture the center of the pulse.

The trigger delay function is an extremely valuable resource in a full power attack.

When the digitiser receives a valid trigger, it waits a specific length of time determined by the value of the trigger delay. Once this time has elapsed it begins to sample the waveform at its input. This means that an attacker can attack each section of the waveform at a time by setting different values for the trigger delay and it eliminates the need for large amounts of memory and processing power. In this case, it was used to capture a section of the waveform that was free from any transient voltages (such as the rising edge of the signal). This way the vertical range wouldn't have to be increased to capture the whole signal and there was less chance of quantisation errors being introduced.

Once this new signal was captured, it was passed through a highpass Butterworth digital filter with a lower cut-off of 200kHz. The purpose of this was to eliminate the 90kHz signal shown in figure 4.25. The resulting signal is shown in figure 4.26. Each exponent operation is clearly visible this time and the time between crests is exactly as calculated theoretically - 72.55μs.

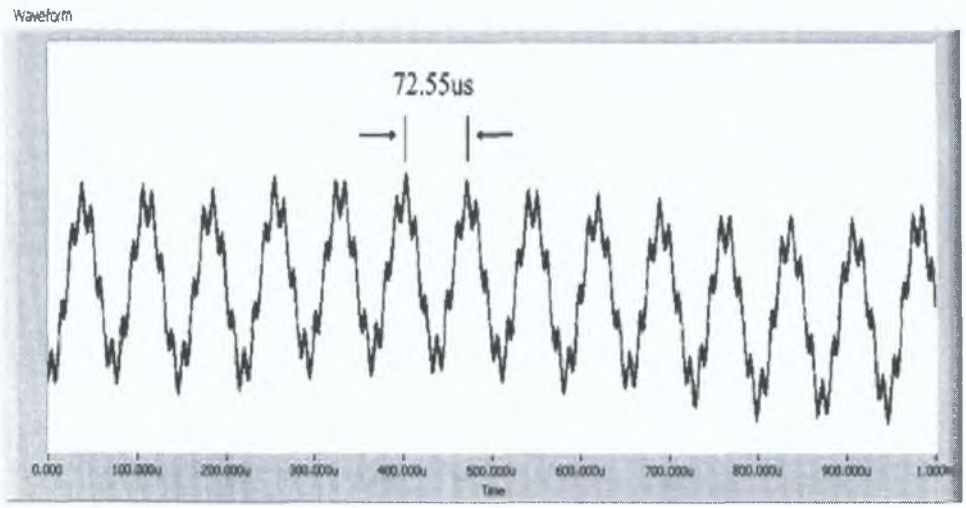


Figure 4.26: Close up of the trace obtained using the all ones key. This particular waveform was collected at the center of the 158ms pulse and was passed through a Butterworth high pass filter with a lower cut-off of 200kHz. Each exponent operation is clearly visible however its value cannot be identified.

Unfortunately these improvements still didn't reveal the secret key and it was clear that some modifications of either the DUT (decrease its clock speed for example) or the equipment was in order. Before doing this however it seemed prudent to check the complete setup with a microntroller designed to emulate the operations on the DUT.

4.3.4 Emulated Setup

The failure of the initial DPA attack was not surprising due to the reasons stated above. Despite this, the exact reason for the failure wasn't known and a number of explanations were possible. Although the simulation had proven that the software was implemented correctly, this didn't account for the rest of the setup. To determine whether or not the entire setup was sound, a circuit was designed that emulated the operations of the DUT. This circuit was a lot less complicated than the DUT but was similar to that used by modern day smartcards. Its main function was to implement the algorithm of figure 2.7[12].

The circuit was based around the PIC16F877 microcontroller from *Microchip Technology Inc.* [83]. This is a 40 pin, 8 bit CMOS flash microcontroller that can be erased and reloaded with a new program as desired. The pin diagram of this device is shown in figure 4.27 for reference. Presently there are quite a number of Microchip's PIC microntrollers on the market (as well as microcontrollers from other manufacturers) however the PIC16F877 was chosen for two main reasons. Firstly, a flash programmer was available for this particular device which saved the expense of ordering in a new one. Secondly (and equally as important), the chip contained 8k $\times$ 14 words of flash memory for storing the software which was more than enough for the application at hand.

[12] Although it was realised that the version of this algorithm was implemented quite differently on the DUT, this wasn't a problem. The purpose of the emulating circuit was simply to prove that the methods and setup were correct.

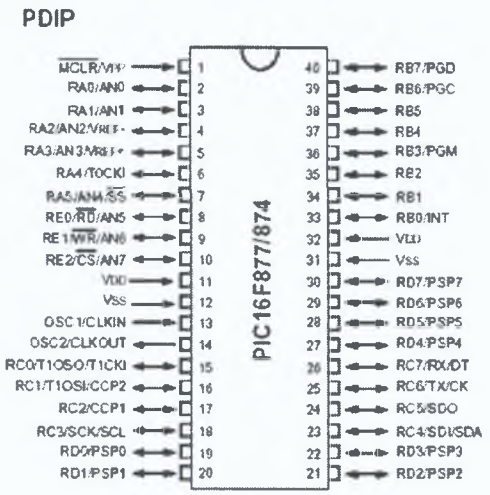


Figure 4.27: Pin diagram for the PIC16F877 microcontroller used to emulate the operations of the DUT. The chip can accept and number of inputs as well as produce a number of outputs (as implied by the two way arrows) which can be used to control the value of the data being operated upon (using a switch) as well as displaying the results (using an LCD display).

Initially, it was necessary to learn the basics of programming the microcontroller. The data sheet was downloaded from Microchip's website to get some familiarity and understanding of the device. A number of simple programs were then written to improve this understanding. Once the basic commands were mastered it was possible to design a circuit to carry out Montgomery's exponentiation.

The program could have been written in either assembly language or a higher level language such as "C". C was chosen because, being a high level language, it required less development time than would have been necessary if assembly code had been used. As well as this fact, a certain level of efficiency in C was possessed which helped reduce this time even further. A special C compiler had to be obtained in order to compile the code. The compiler used was known as *C2C*. This was a fairly basic compiler in the sense that it could only handle integer type variables. Fortunately, this was not a problem as number types other than integers do not exist in modular arithmetic.

What was desired from this circuit was proof that the existing experimental setup was sound and that the value of the exponent could be determined by measuring the power fluctuations. The particular attack chosen was the MESD attack (described in section 2.5.4.1) as it didn't require inputing random data. Although C contains a function for producing pseudorandom numbers, this function is far from ideal and true randomness is very difficult to achieve. Also, the MESD attack was the attack of choice against the DUT as it wasn't possible to send random data to this device either.

The MESD attack was one of the most powerful attacks against the modular exponentiation scheme as reported in [27] so its failure against the microcontroller would have suggested that a problem existed. However, if it succeeded in revealing the secret key of the microcontroller but failed against the DUT (as was the case) this would increase the level of confidence in the security of the DUT[13]. It was unlikely that the other attacks described in [27] would have been successful had the MESD attack failed.

The attack was carried out one bit at a time. The bits before that being attacked were assumed to have been successfully compromised and the value of the key up to that point was therefore known. Each attacked bit could take on two possible values and the values of the bits after it were unknown and were set to some random value. In order to attack each bit, the PIC was loaded with the two possible values of the key. The particular key used was determined by a switch connected to the microcontroller. For example, to use the key value where the guess of the bit was a 0 the switch would have to have been in the "OFF" position whereas to use a guess value of 1 it would have to have been in the "ON" position.

The PIC was run at a clock speed of 4MHz using a quartz crystal for optimal stability. This frequency was chosen because it allowed the digitiser to collect enough sample points without having to use its RIS mode thereby decreasing the processing time

[13]However it must be pointed out that the reduced clock speed of the PIC increased the possibility of a successful attack for this particular setup.

from hours to minutes. Smartcards are generally run at approximately 3.57MHz so this allowed it to closely emulate a normal smartcard also.

It was desired that the result of the modular exponentiation be displayed to verify the circuit was functioning correctly. This was achieved using a HD44780 based Liquid Crystal Display (LCD)[14]. This device proved quite difficult to program initially but worked as expected once the correct handshaking procedures had been setup between the PIC16F877 and the LCD (along with time delays to allow the LCD time to complete one task before being sent a second). It was possible to send the LCD data either one byte at a time or in two four bit nibbles. The second option was chosen simply because it involved less connections and provided room for other external components if needed.

Once the circuit was fully operational, an 18Ω resistor[15] was placed in series with the ground line to allow measurement of the power fluctuations. It was possible to place this resistor in series with either the power or ground line but the latter was chosen because the digitiser (as with all oscilloscopes) measured the voltage at a particular node with respect to its own ground (which was connected to the true ground of the circuit). Had the resistor been connected across the power supply, two probes would have been needed for the same measurement - one for each side of the resistor. It would then have been necessary to calculate the difference between the two waveforms and although an option was available to allow this calculation, it would only have complicated the measurement and increased the processing time for an attack. It might be considered that the probe tip could have been connected to one side of the resistor and the ground connection to the other side however this would only have shorted the supply as it would then have been effectively connected directly to ground.

With the resistor in place and the probe connected, the PIC was switched on and the

[14]The actual device used was a Powertip PC 1602-F available from Farnell electronics [84].
[15]The small value was chosen so as not to adversely affect the circuit.

Labview program run. The value of the "secret" exponent was simply 1111^{16}. The experiment began by attacking the third bit. It was assumed that previous attacks had been successful and that the key up to this point was known. At this stage therefore, the known key value was 11??. A guess was taken at the third bit and the last bit was set to a zero. Under normal circumstances the value of the remaining bits of the key would be set to a random value, however for this experiment it was desired that they be different from those of the "secret" key. For ease of operation, the PIC was programmed with both 1100 and 1110, the chosen value being determined by the position of a switch as discussed above. The program was set to run in an infinite loop (once a key was chosen) with one of the output pins (RA0, see figure 4.27) being set to go high at the beginning of the modular exponentiation and low at the end. This allowed for a fairly steady trigger signal.

Initially the value 1100 (an incorrect guess) was tested. The results of this MESD attack are given in figure 4.28 where it can be seen that their is no difference in the power fluctuations for the secret and guessed keys up to the unknown third bit. This is due to the fact that the bit was incorrectly guessed. Had this bit been correctly guessed, the power signals would have been the same up to the fourth bit showing a value of zero in the difference signal up to this point. The waveform for the correct guess is seen in figure 4.29. Both of these signals can be compared to the results reported by Messerges et al. in [27]. The results of the two differential waveforms from this paper are shown in figure 4.30 for reference.

An interesting point to note is that the waveforms of figures 4.28 and 4.29 were taken with absolutely no amplification whatsoever. A standard coaxial probe ($\times 1$) was used in the collection of the signals. The number of waveforms collected for each average was 2000 although this was far more than was necessary. Experiments showed that the

[16] Although this number is a lot smaller than that used in normal cryptographic systems, this didn't matter as the same operations are simply repeated for every bit.

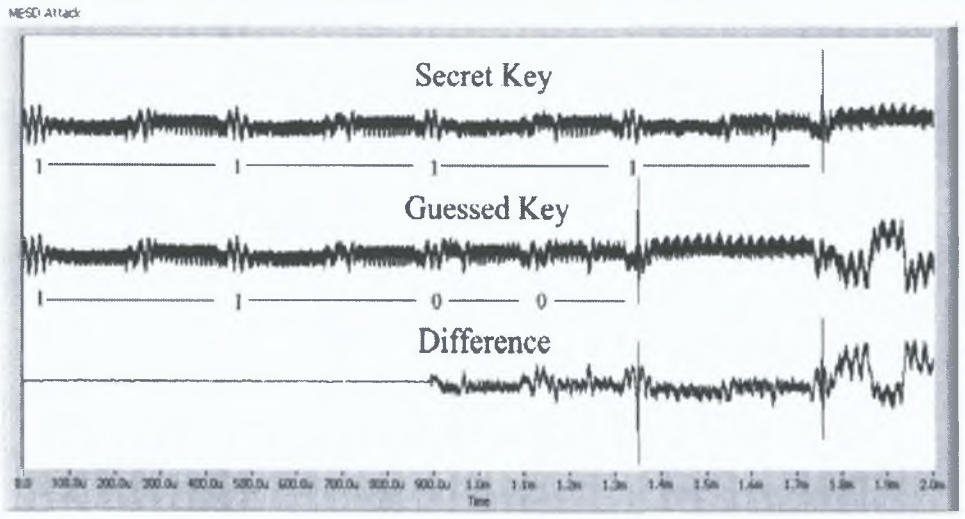


Figure 4.28: The difference between the average power of the secret key and that of a key with an incorrectly guessed bit. The values up to the attacked bit had been successfully retrieved and the difference in power fluctuations tend to zero up to this point.

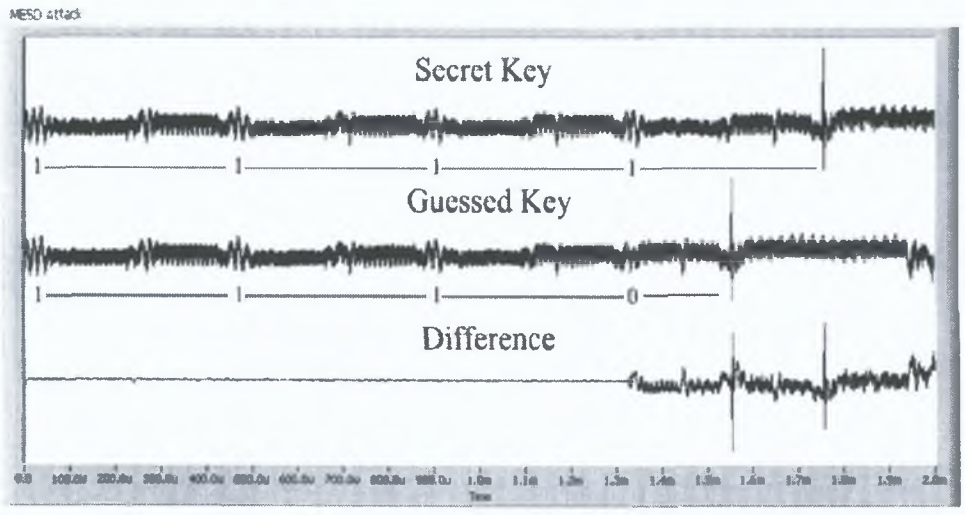


Figure 4.29: The difference between the average power of the secret key and that of a key with a correctly guessed bit. It can be seen quite clearly from the differential trace that the waveforms are now the same up to the fourth bit.

difference between the two was noticeable after only 50 averages of each waveform. This was even lower than the results reported by Messerges et al. in [27] where at least 200 waveforms were needed for a successful attack.

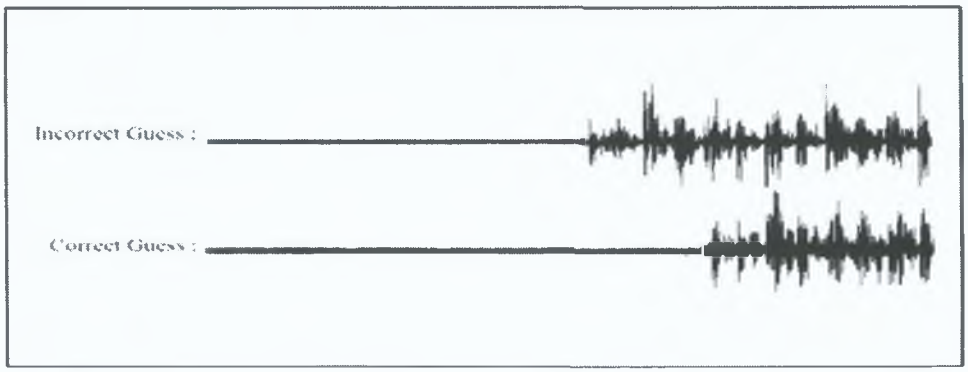


Figure 4.30: Results of the MESD attack reported by Messerges et al.. It can be seen that these waveforms are very similar to those shown in figures 4.28 and 4.29.

Closer examination of the waveforms in figures 4.28 and 4.29 shows that the time it took to operate on a 1 was a lot less than that of a 0. This was to do with the fact that the multiply operation was not performed if the value of the bit was a 0. A magnified view of the waveform taken for the exponent 1111 is shown in figure 4.31. It can be seen here that each 1 bit contained two distinct sections. The first section is the power fluctuations caused by a multiply operation whereas the second section is that caused by a squaring operation. In the case where the exponent was a 0, only the squaring operation was performed. There was a distinct difference in both time and power between these two sections of the waveform. The time difference can be seen with the large spikes at the end of each exponentiation. These spikes are the result of coupling from a separate pin which was set to go high at the end of each exponentiation. This difference in time would not only have left this particular implementation open to a power attack but also a timing attack. In this case the difference in the time it takes to perform a particular operation is used to learn details about that operation (see [22, 85–87] for more details). A close up of the multiply and square sections of the waveform is given in figure 4.32 where the difference between the two sections is exemplified.

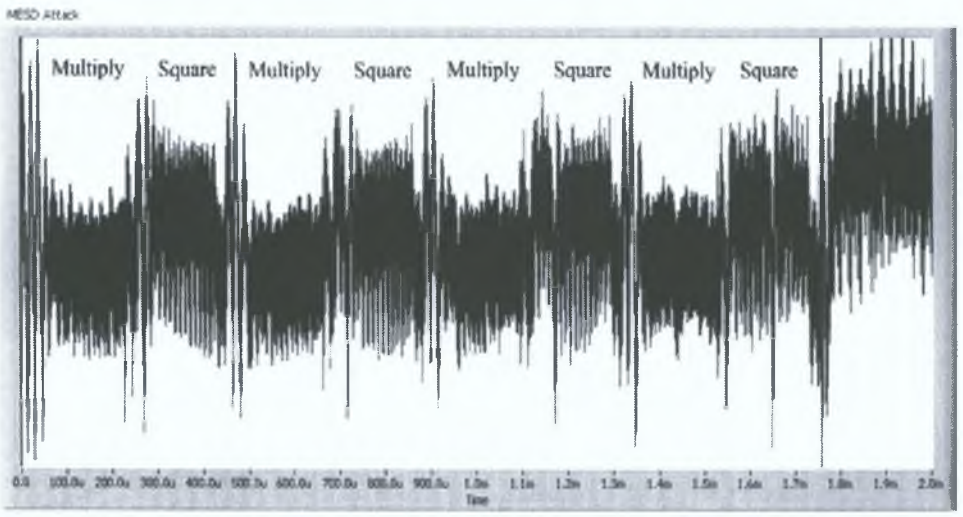


Figure 4.31: A magnified view of the power waveform obtained with the exponent 1111. The differences between a square and a multiply operations are clearly visible.

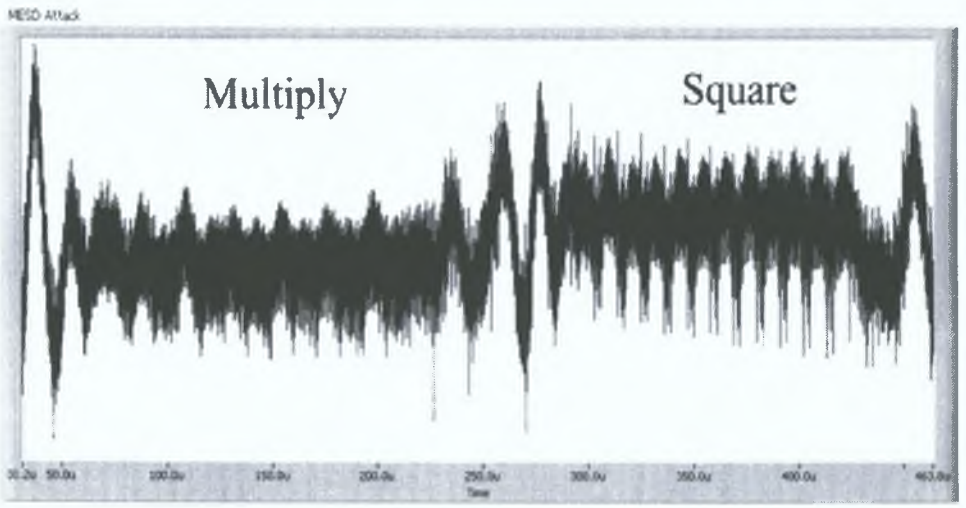


Figure 4.32: The power waveforms representing the square and the multiply operation. The two sections are clearly distinguishable. The multiply operation took slightly longer to complete than the square operation.

As a preventative measure against these power attacks it was decided to perform the multiplication in every round and to only update the register if the bit in the exponent was a one (see figure 2.7). The results are shown in figure 4.33 where it can be seen that the differential trace has been reduced in amplitude but is still clearly visible. The

processing times for each bit are almost the same but there is still a slight difference. This can be seen by the near alignment of the spikes at approximately 1.7ms and can be compared to that in figures 4.28 and 4.29 where the time for each encryption is quite different.

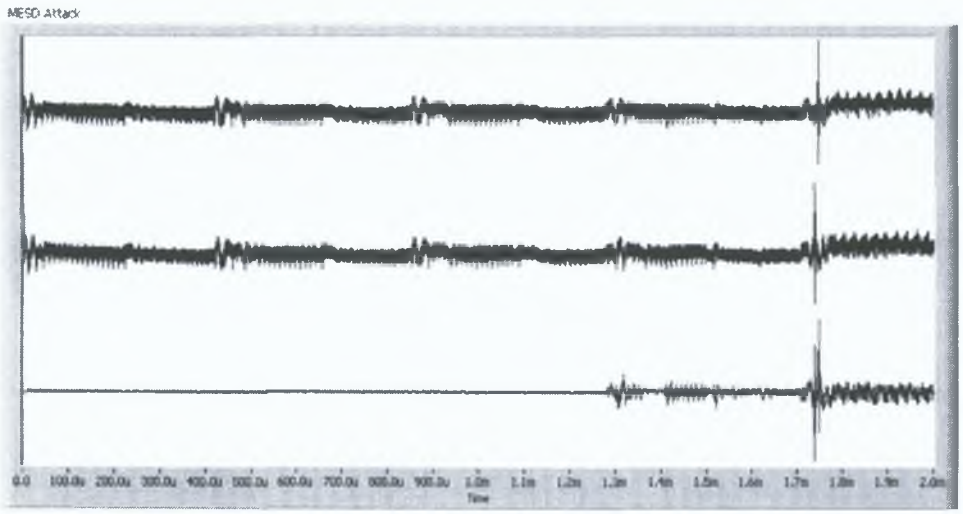


Figure 4.33: The power waveforms and differential trace when the multiplication operation is performed in every round. There is still a slight time difference and the difference waveform is still correlated to the key.

Another test was to insert a second "if" statement into the program. This time however, the condition tested was a 0 and not a 1 (again see figure 2.7). The body of this if statement contained the exact same multiplication operation as the first one however the results were saved to a dummy register thus the final result was unaffected. It was hoped that the length of time for each operation would now be identical and that the only chance of the attack succeeding was if the data dependent power fluctuations contained differences.

The result is shown in figure 4.34 where it can be seen the length of time it takes to process each exponent is the same. The differential trace has reduced slightly in amplitude again however it is still visible.

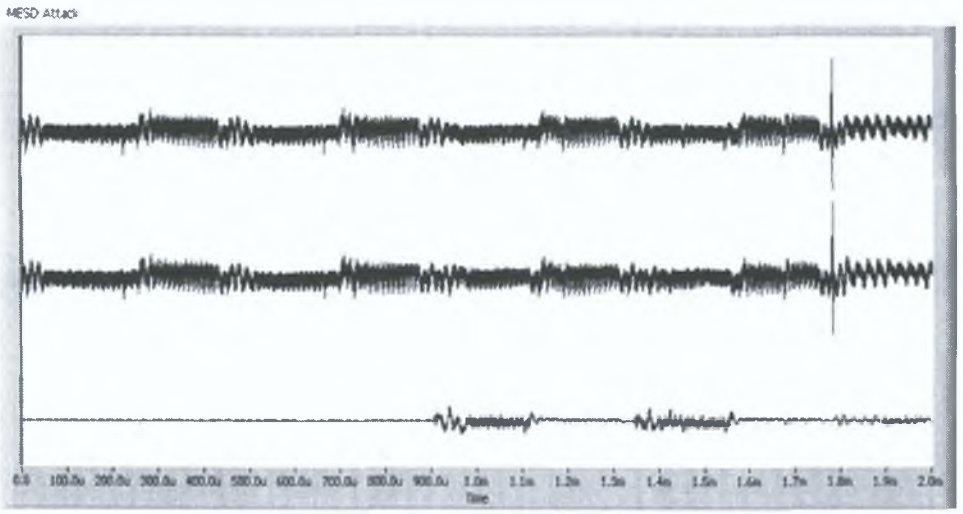


Figure 4.34: PIC run with two if statements. One for a 1 and the other for a 0. The difference is still distinguishable.

As a final attempt at a solution, it was decided to implement both of the above operations and to see how the differential trace is effected. The meant that the multiply operation and a register update were performed every round, regardless of the value of the exponent. The results are shown in figure 4.35 where it can be seen that there is now even less difference in the timing of the two waveforms. The differential trace has been reduced in amplitude from the previous cases however it is still visible. This shows the true power of the MESD attack. It doesn't rely on the observations of single bias peak in a waveform but allows a much clearer view of the information.

The graphs presented here show without a doubt that the experimental setup was sound within the limits of the equipment. It also showed the power of the MESD attack and the reason why it is suspected that the DUT may be secure from this attack with equipment used here but would not be secure with an improved digitiser and a better amplification and filtering stage.

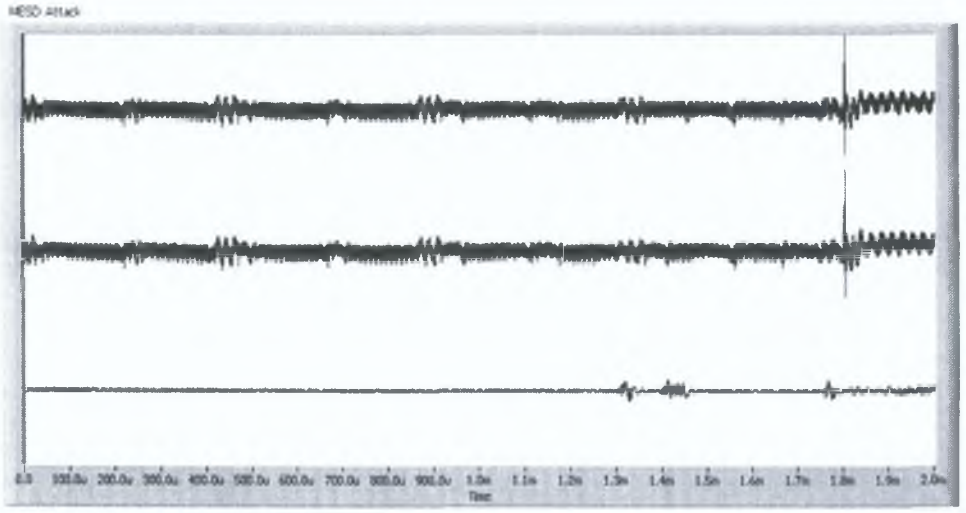


Figure 4.35: These are the power waveforms from a register update only but with an if statement for both the zero case and one case.

4.3.5 Final Power Attack

Having verified the setup using both the simulation and the emulation it was now necessary to return to the DUT and to attempt to improve the attacks. A number of things were done to accomplish this.

- The clock frequency of the DUT was reduced to 30MHz.
- A steady trigger signal was extracted from the board (allowing the RIS mode to be used and thus increasing the effective sample rate).
- A flash programmer was obtained that would allow the chips to be programmed with any value desired.
- Two new high bandwidth (80kHz - 10GHz) amplifiers were obtained.

These adjustments were great improvements to the experimental setup and a more realistic DPA attack could now be attempted.

4.3.5.1 SPA

With the clock rate reduced to 30MHz (a value that was more likely to produce results with the available equipment), the same setup as that shown in figure 4.15 was used to capture the new waveform. The result is shown in figure 4.36 where the length of the pulse had now increased to

$$\begin{aligned}\frac{4353}{30 \times 10^6} \times 2176 &= 2 \times 72.55 \times 2176 \\ &\approx 316\text{ms}\end{aligned}$$

and the time for each multiplication has increased to

$$\begin{aligned}\frac{4353}{30 \times 10^6} &= 2 \times 72.55 \\ &= 145.10\mu s.\end{aligned}$$

The agreement between the updated equations and the updated measurements confirmed exactly what the waveform represented.

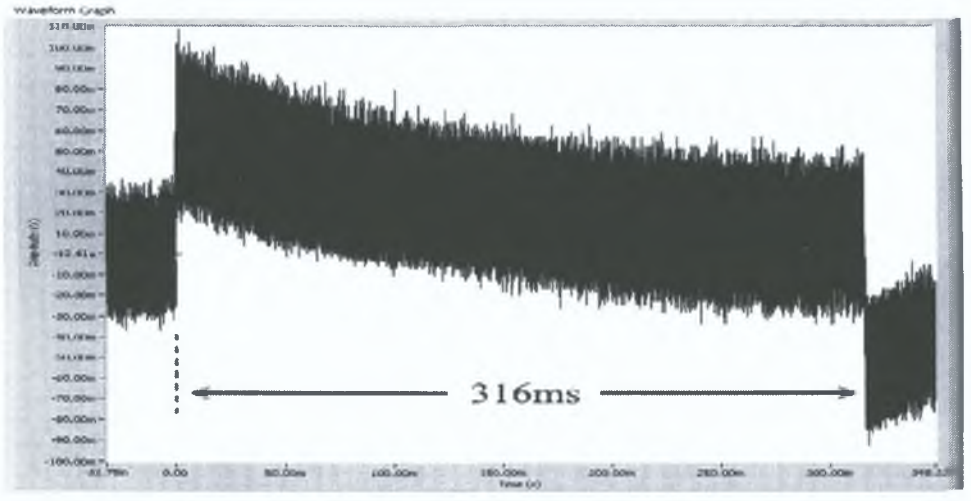


Figure 4.36: This is the pulse representing one complete encryption. As the clock had been reduced to 30MHz, the operations took twice as long to complete and the pulse width was now twice that of figure 4.16.

Along with the reduction in clock frequency a new trigger signal was obtained and is shown in figure 4.37. This signal was retrieved from the pin of another chip on the board that was set to go high at the beginning of each encryption. As can be seen from the figure there are five pulses each of a different duration. Because of this it was feared that the digitiser might trigger off a different pulse each time. As a preventative measure, the *holdoff* option on the NI5112 was used. This option ignored any signals (that might have been mistaken as a trigger) for a specified time period after the acquisition of a valid trigger. The particular time period used was 1ms however any time greater than about $100\mu s$ would have sufficed as the pulses lasted for a duration of $57.5\mu s$. The first trigger pulse coincided exactly with the beginning of the encryption operation.

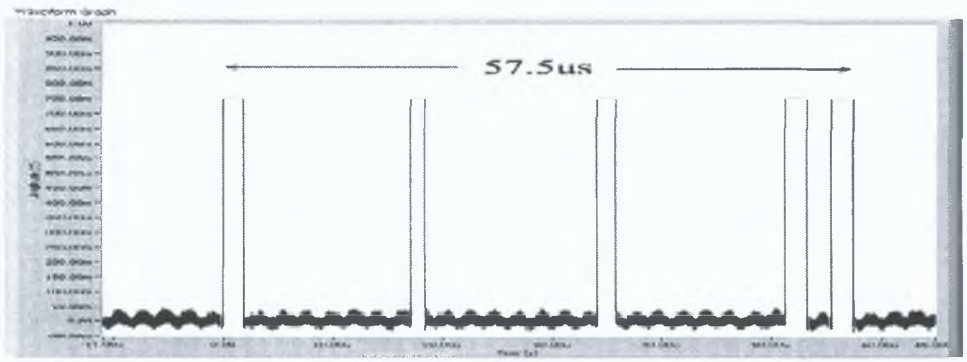


Figure 4.37: Trigger signal used to allow a steady capture of the power waveform. The encryption began at the beginning of the first pulse. It was necessary to set the "holdoff" option on the digitiser to 1ms so as not to accidently trigger off the other four pulses.

Using the trigger signal, it was attempted again to use the RIS mode of the digitiser. Unfortunately, even with the stable trigger it still wouldn't work. National Instruments support were contacted where it was explained that the digitiser expected an analog signal as a trigger for RIS mode and that the fast rise time digital signals that were available wouldn't suffice. This seemed rather strange and was extremely unfortunate as one of the reasons this particular digitiser was purchased was because of its RIS

capabilities.

Some methods were initially devised to try to overcome this problem. It was decided to build a low pass filter that would smooth out the rise times of the signal. This filter was designed in PSPICE and its output was initially thought to be adequate. However after re-examination of the trigger signals, it was noticed that the widths of each of the pulses were not constant on every encryption. This caused the filtered signal to rise and fall in accordance with the pulse width. As a result, this signal wasn't stable enough to trigger off.

The next idea was to use a signal generator to accept the pulses of figure 4.37 as a trigger and produce a smooth output. This was more of a success than the last method however it was still feared that the output wasn't stable enough for an attack[17]. It was decided at this stage to try to have another look at the subvi that controlled the RIS trigger configuration. After some extensive analysis a problem was found at a fairly low level of the program that was preventing the pulse waveform from being accepted as a valid signal. This problem was corrected and the waveform was accepted as a valid trigger. It was unfortunate that the trigger signal was initially considered unsuitable as this wasted quite a bit of time in attempting to remedy it.

With RIS finally working, it was possible to effectively sample the waveforms at a maximum rate of $2.5GSs^{-1}$. This was quite adequate for the application at hand and it eliminated the need for an antialiasing filter as the 100MHz analog bandwidth of the digitiser prevented any signals above 1GHz (half the sampling frequency) from passing. Unfortunately RIS mode had two major shortcomings. The first was the length of time it took to collect each waveform. This time increase was extremely prohibitive and caused the overall acquisition time to increase from hours to days[18].

[17] Although RIS worked with this signal its stability was paramount. If it wasn't stable, the sample points would have been misaligned on each run of the algorithm producing erroneous results.

[18] RIS wasn't the only reason for this increase in time. The higher sample rates meant that the number of points required needed to be increased from 150,000 to 3,000,000 just to collect 1.5ms of data.

The second shortcoming was the fact that the trigger delay option was not available in RIS mode. This was quite a bit of a problem as it meant that only the initial part of the waveform could be analysed. This section of the waveform was corrupted slightly by the transient voltage that existed at the beginning of each encryption (i.e. the rising edge of figures 4.16 and 4.36).

Another major improvement to the setup was the provision of a flash programmer. It was then possible to choose any value for the base, the modulus and the exponent[19]. As it was not possible to use the trigger delay option in RIS mode, it really wasn't possible to do a full power attack as different sections of the waveform could not be attacked. However, if even one bit could have been revealed, then this would have shown that the board was vulnerable to a full attack with some slightly better equipment.

It was decided to reduce the exponent to a more manageable value of 16 bits (the base and modulus were left at 2174 bits at that stage). The values used were

$$e_1 = 1111\ 1111\ 1111\ 1111$$

$$e_2 = 1000\ 1010\ 0100\ 0111$$

The hardware was set to detect when the leading bits of the exponent were all 0 thus signifying its end. It was necessary therefore to set the MSB of each exponent to 1 in order for the waveforms to be the same length. For example, assuming the MSB (left most bit) of e_1 was a 1 then the key would be 16 bits long and the time for a complete encryption would be

$$145.10 \times 10^{-6} \times 16 \approx 2.3\text{m}s.$$

However, if the MSB of e_2 was set to 0 then the time for a complete encryption with

[19]However it still wasn't possible to automate this choice and therefore random values couldn't be produced in any reasonable time frame.

this key would be

$$145.10 \times 10^{-6} \times 12 \approx 1.7\text{m}s.$$

Had the MESD attack been carried out with these values it would have appeared to have been successful and the result would have looked similar to that of figures 4.28 and 4.29 however these results would have been erroneous.

Results were taken again using these new exponent values. Figure 4.38 shows two full encryptions. The waveform was captured using a $\times 10$ probe and no amplification. Comparing this with figure 4.17 it can be seen that there is quite a difference. As before, the time between two consecutive encryptions is 812.5ms however the pulse widths have changed dramatically. Each encryption takes about 2.5ms (the 2.3ms calculated plus possibly some extra overhead for loading registers etc). Due the entire bandwidth of the digitiser being used, there was an excessive amount of noise. As a result the peak amplitude of the pulses appeared as if they are just over 80mV however they were closer to 45mV in reality. Figure 4.39 shows the same graph with the 20MHz bandwidth limit on. It can be seen now that the pulses have reduced to about 70mV.

The next step was to magnify each pulse to see what information could be retrieved. Figures 4.40 shows the pulse taken at the maximum realtime sampling rate of 100MSs^{-1}. The 20MHz bandwidth limit of the NI5112 digitiser was turned on to prevent aliasing. It can be seen that the pulse was almost the same length as that calculated and that each individual operation was quite clearly visible (however it was still possible to mistake this for the surrounding noise). By increasing the sampling rate using RIS the signal became a lot clearer and each exponent operation could be identified with little ambiguity as can be seen from figure 4.41. It can also be seen that the trigger signal coupled over onto the pulse and was made visible by the higher sampling rate.

Having reduced the key and observed the reduction in the width of the pulse, it was

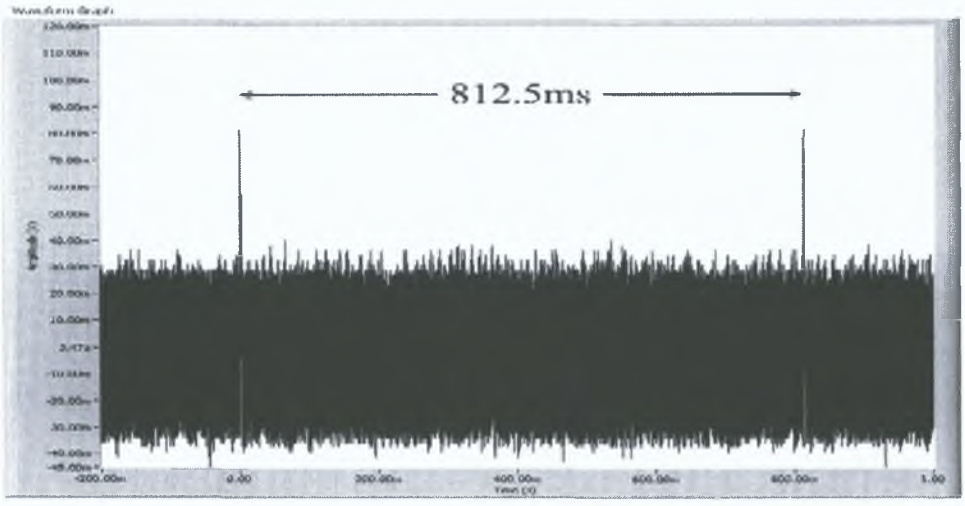


Figure 4.38: This graph shows two complete operations. Each operation lasts about 2.5ms compared with the 812.5ms between operations. The pulses representing the encryptions are at a level of just over 80mV due to the excessive noise.

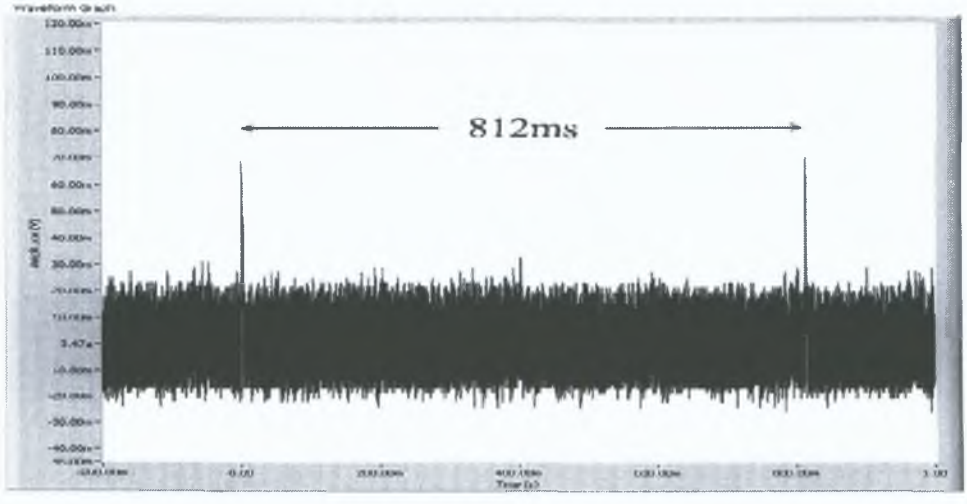


Figure 4.39: This graph again shows two complete operations however this time the bandwidth has been limited to 20MHz. The noise dropped quite substantially and the pulse appeared to be only 70mV in amplitude.

decided to reduce the base and modulus to see what effects they would have on the power trace. The first change was to make all the bits of the modulus equal to 1, the base equal to one less than the modulus and the key equal to any odd value. Using

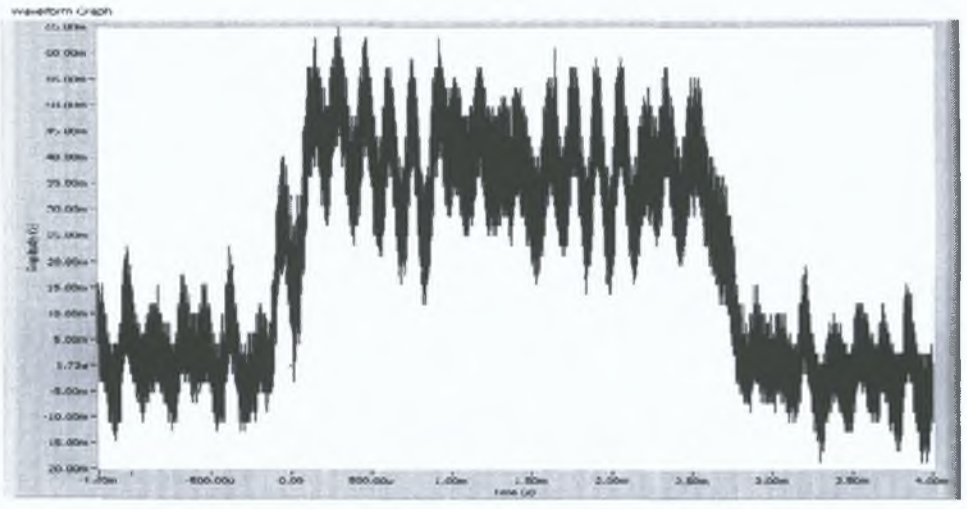


Figure 4.40: One complete encryption. This was sampled at $100MSs^{-1}$ and was limited to 20MHz bandwidth to prevent aliasing. Each of the 16 exponent operations could be seen with some clarity however it was possible to mistake them as background noise.

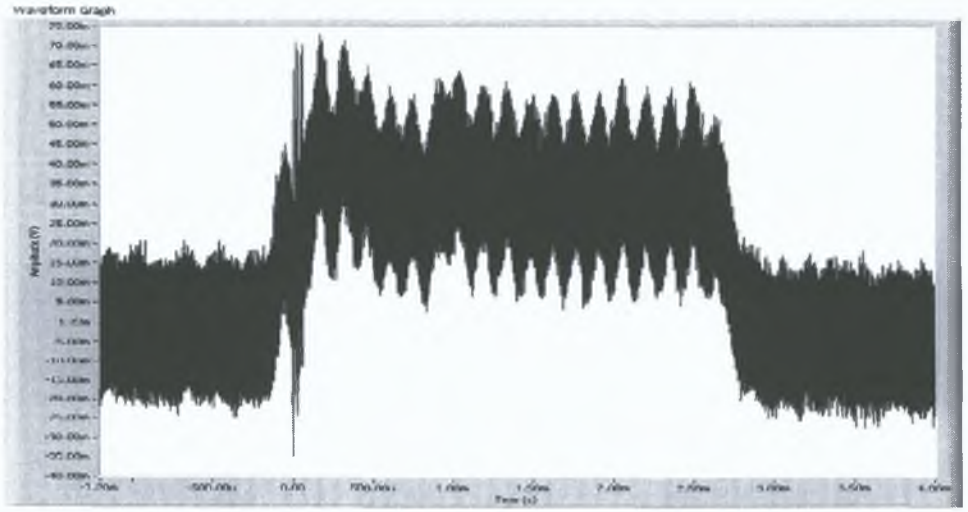


Figure 4.41: This particular encryption was sampled at an effective rate of $2GSs^{-1}$ using RIS. The 16 exponent operations are a lot more distinguishable in this case.

these values and the following characteristic of modular exponentiation, it was known

that the result would equal to the base:

$$(b-1)^e \bmod b = \begin{cases} b-1 & \text{for } e \text{ odd,} \\ 1 & \text{for } e \text{ even.} \end{cases} \tag{4.3}$$

As the modulus was one less than the base the result therefore contained all ones (except for the LSB which was a 0). The power waveform was then captured and the result is shown in figure 4.42. As can be seen, the pulse waveform has completely disappeared. This initially appeared very strange but some extra experiments showed that in fact it was consistent with the theory described in chapter 2.

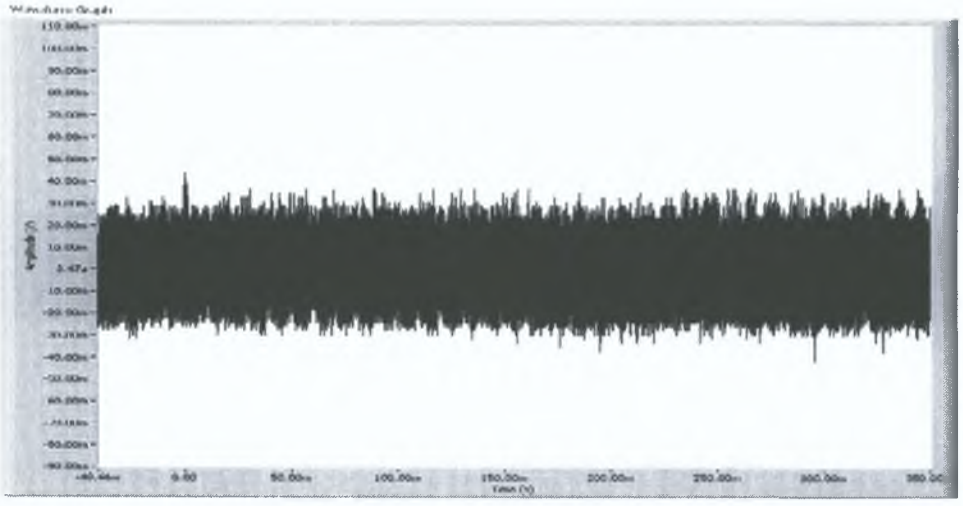


Figure 4.42: Waveform collected for a modulus of all ones and a base one less than the modulus. The key is some odd value.

The next change made was to the key which was set to an even value. This caused the result to equal 1 as determined by equation 4.3. The power waveform was identical to figure 4.42. Therefore it was seen that the waveform was the same for the result being all zeros and all ones (the LSB being a 1 would have had very little effect and was ignored). At this stage a pattern can be seen - if all the bits are the same very little power is used.

The next thing that was tried was to change the last two bits of the base to 1001 1001 so equation 4.3 would no longer hold. This would mean that the base was practically all ones and the modulus all ones however the result would be some pseudorandom combinations of ones and zeros. The power waveform that resulted is shown in figure 4.43.It can be seen that the pulse has reappeared. Before attempting an explanation two final tests were carried out.

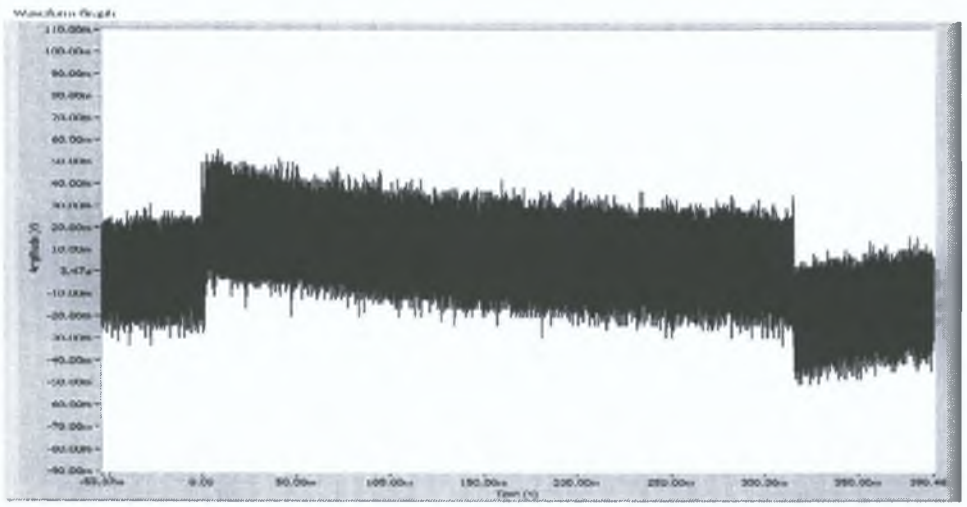


Figure 4.43: The two LSBs of the base are changed to 1001 1001 so the equation 4.3 doesn't hold and the result appears random.

The base was set to alternating (1010 1010 . . . 1010) and the modulus set to the value of the base plus one. The key was set to an even value which produced a result of practically all zeros (i.e. 0000 0000 . . . 0001). The resulting power waveform is shown in figure 4.44. This waveform was slightly unexpected. It had the largest amplitude out of all the values tested so far. Although a certain pattern was emerging, it wasn't fully what was expected.

The key was then changed to an odd value. This caused the result to equal the base. The resulting power waveform is shown in figure 4.45. This was a rather strange result as it was expected that this value would be greater than the previous. However a hypothesis

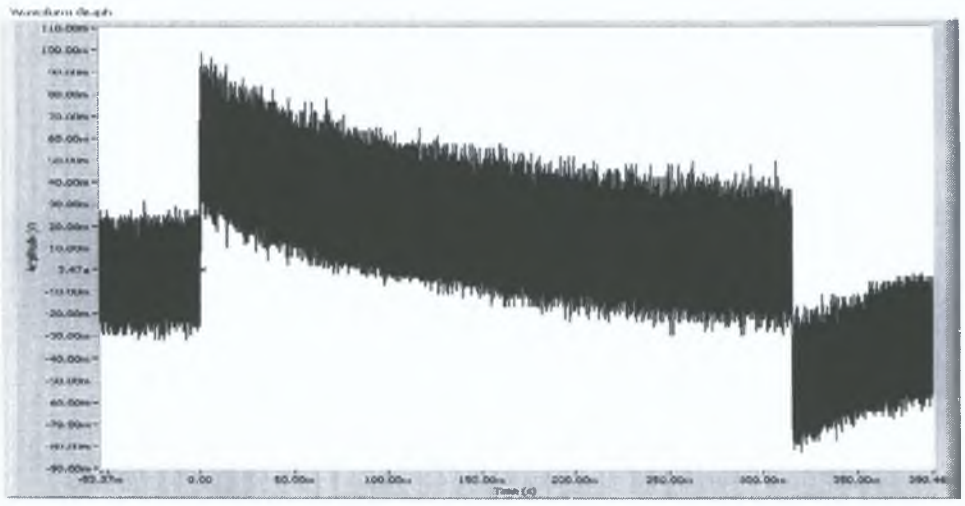


Figure 4.44: Base and modulus set to alternating ones and zeros and the result is 1. This waveform was the largest one obtained.

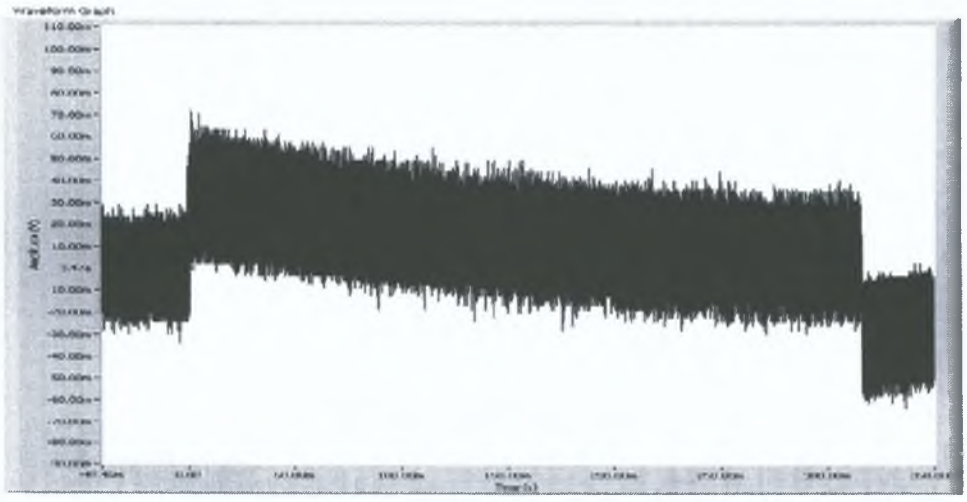


Figure 4.45: Base and modulus set to alternating ones and zeros and the result is equal to the base. The waveform has reduced in size.

could now be put forth.

From the theory in chapter 2, it was known that current was only drawn when the logic gates switched from one value to another (disregarding the leakage current). It was seen from the first two tests that when the base and the output were the same

value (regardless of whether they were 1's or 0's), no noticeable power fluctuations were present. However when the base consists of alternating ones and zeros and the output was all zeros, the pulse had its maximum amplitude. Alternating ones and zero's were chosen because this would cause all the transistors to switch thereby drawing the maximum amount of current. However, when the result was set to alternating zeros the pulse amplitude was less. It was clear then that the result somehow counteracted the amount of power drawn by the base which suggested that it reduced the number of logic gates that were switched. Unfortunately not enough information was possessed about the board to confirm this hypothesis.

Although SPA provided a lot of information, it did not reveal the secret key in any way. The next thing to try was a full DPA attack which had the promise of success due to the upgraded setup. However as will be seen, this was not to be.

4.3.5.2 DPA

The setup of figure 4.15 was once again used in the final DPA attack. Initially only the EMC amplifier was used so as not to filter out the lower frequency content. The key was set back to the values e_1 and e_2 given above, and the base and modulus set to some random values (but left constant throughout the experiment). It was necessary that the modulus be odd for reasons discussed in Appendix B. RIS was used and the digitiser was set to sample at 2GSs^{-1}. This value was a compromise between lengthy processing times and misalignment of the sampled data[20]. To capture a 1ms portion of the power waveform the number of points required was

$$2\text{GSs}^{-1} \times 1 \times 10^{-3} = 2 \times 10^{6}$$

[20] As discussed in [76] the lower the sample rate the more possibility of the sample points of an asynchronous attack being misaligned.

sample points.

The portion of the waveform that was of interest was captured using the digitiser. Theoretically the time slice that was required was

$$145.10 \times 10^{-6} \times 4 = 580.4\mu s$$

due to the fact that the exponents differed after three bits. However, a value of 1ms was captured instead to ensure that the correct portion of the wave was being analysed.

The Labview program was run 1,000 times for each key. As mentioned the length of time to complete this was prohibitive and prevented a lot of different ideas being tested. It took exactly four days to collect the data for the two exponents and the samples took up about 22GB of hard drive space. When the 1,000 waveforms were averaged they were stored into a single file and the experiment run again. This allowed 1,000 averages to be taken at a time.

After eight thousand averages the results were processed to produce an average of two waveforms and a differential trace. The two averages can be seen in figure 4.46. The apparent difference in offset was manually put in to display the waveforms on the same graph and was not an effect of the different keys. The captured waveforms were amplified using the EMC amplifier which allowed the lower frequency components to be retained. It can be seen from this figure that the time difference between the crest of two waves is exactly $145.10\mu s$ as calculated above.

Figure 4.47 shows a close up of the waveform of figure 4.46. The close up is around about the $322\mu s$ area and is well before the waveforms should change. As can be seen, the waveforms are almost identical to each other. This would seem to show that the RIS was performing correctly as these two waveforms were taken at completely different times and only began to resemble each other in this manner after about 2,000

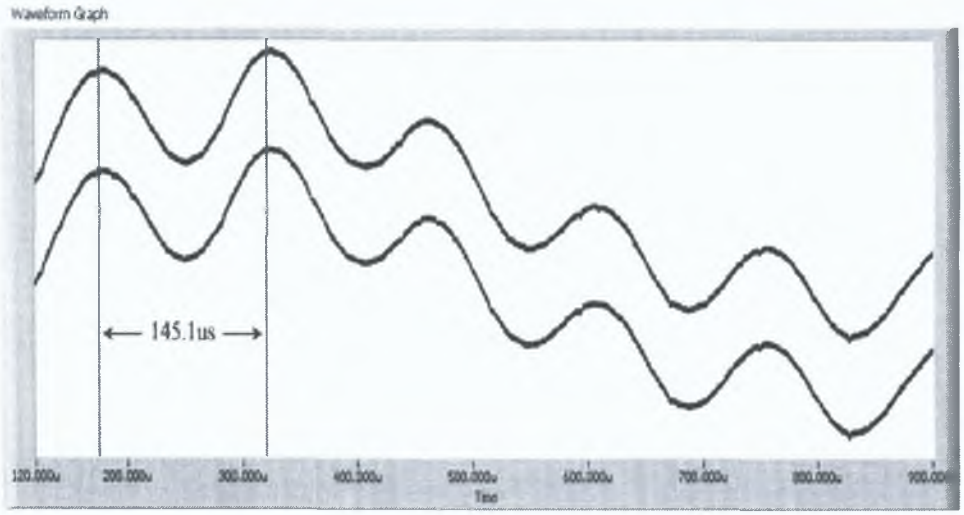


Figure 4.46: Waveforms obtained using the key consisting of all ones (top) and key with half ones half zeros (bottom). These were amplified using the EMC amplifier and the low frequency components are present acting as a guide to the occurrences of each operation. It can be seen that time difference between the crest of each pulse is exactly $145.10\mu s$ which is the time it took for each bit to be operated upon.

averages.

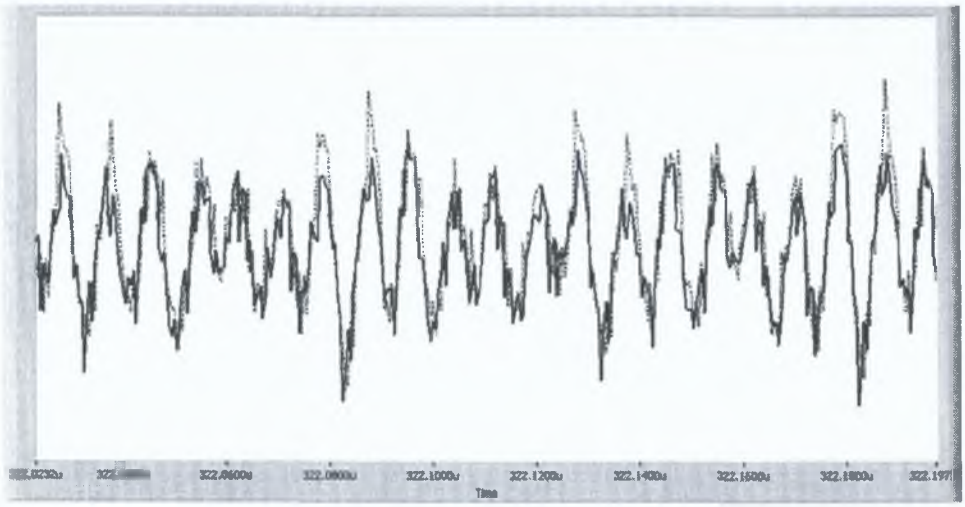


Figure 4.47: This is a close up of the waveform shown in figure 4.46 before the keys begin to differ. The dotted line represents the waveform produced with the all ones key and the continuous line is that produced by the key with half one, half zeros. As would be expected the waveforms are almost identical.

It was then expected that a close up of the waveforms after about $600\mu s$ would show the waveforms to be different. Figure 4.48 shows the result of this where the windowed area is around the $760\mu s$ point. Although the waveforms were still very similar, there was a slight offset between them however the reason for this was not clear. It may have been dependent on the value of the key however no information could be drawn from it. The fluctuations themselves were still the same for both waveforms.

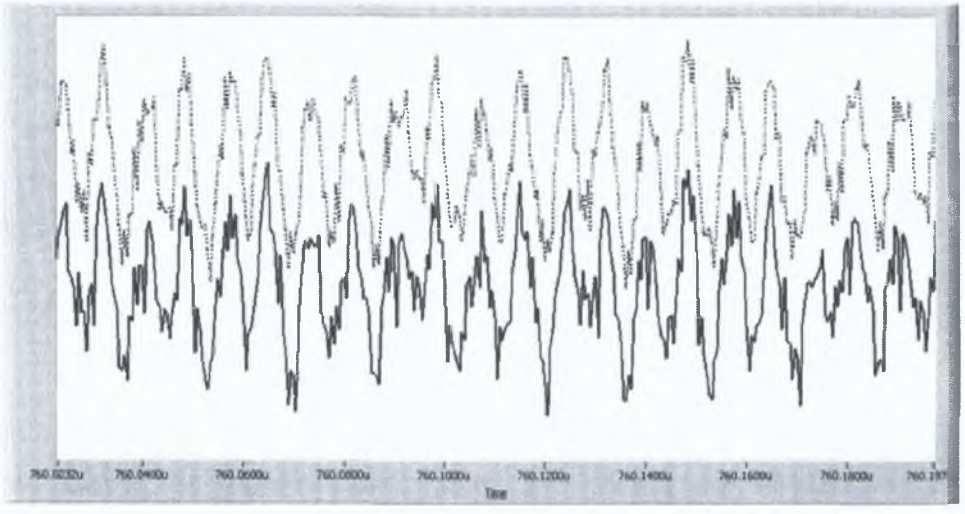


Figure 4.48: This is a close up of the waveform shown in figure 4.46 after the keys have differed. It was expected that the two waveforms be quite different from each other due to algorithmic noise etc. however this appeared not to be the case. The main difference is a slight offset of the two waves however it is doubtful that this reveals any information.

A strange occurrence can be seen on zooming out of figure 4.48. At certain points on the waveform there appears to be 20MHz beating occurring (the frequency was verified through spectral analysis). This can be seen in figure 4.49. This signal may have been caused by the RIS sampling. If the input frequency is a subharmonic of the sampling frequency then it will not be sampled correctly as the samples which are supposed to be randomly distributed, will in fact be bunched together. It is clear that the 20MHz signal is 1/100th of the sampling frequency however as the board is running at 30MHz it is unclear where this signal came from. It is likely that it doesn't represent the operations

running on the DUT.

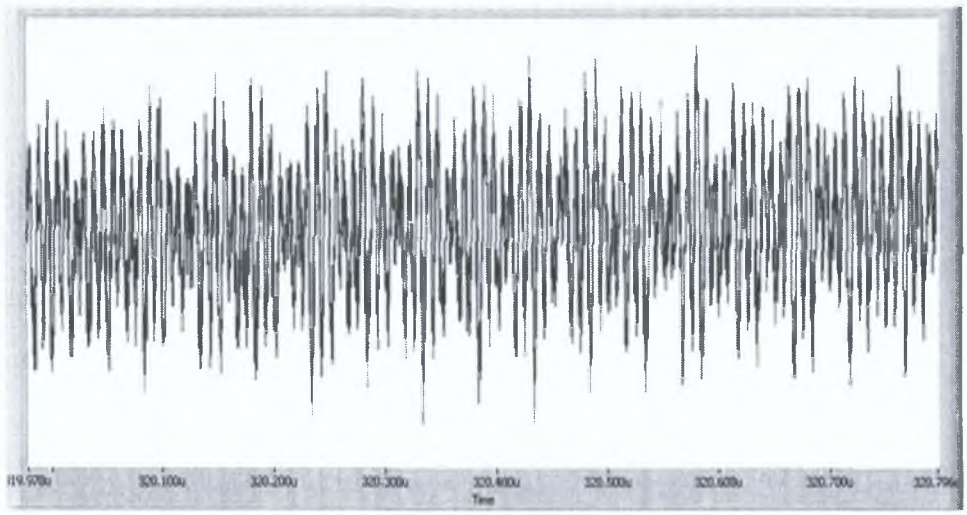


Figure 4.49: 20MHz beating, probably caused by the fact that the RIS was run at 100 times this value.

To get rid of any periodic signals that may not have been reduced by averaging (i.e. to "demodulate" the power waveform and recover the "baseband" or secret information), it was necessary to obtain a differential trace. This can be seen in 4.50. No difference is observed between the first and second halves of the waveform. It was expected that this waveform have a similar form as the differential traces shown in figures 4.28 and 4.30 however this was not the case.

It was unclear why no difference could be observed and it was very possible that RIS was at fault but it may simply have been to do with the fact that not enough averages were taken. Unfortunately due to the length of time it took to collect the power waveforms it was not possible to increase the number from 8,000. Another possibility was the coaxial probe used to measure the signals. It was initially suspected that there may have been some form of filtering or resonance taking place that affected the results however a number of other probes were tested and similar graphs were obtained each time. If the probe wasn't at fault there was also a possibility that the EMC amplifier

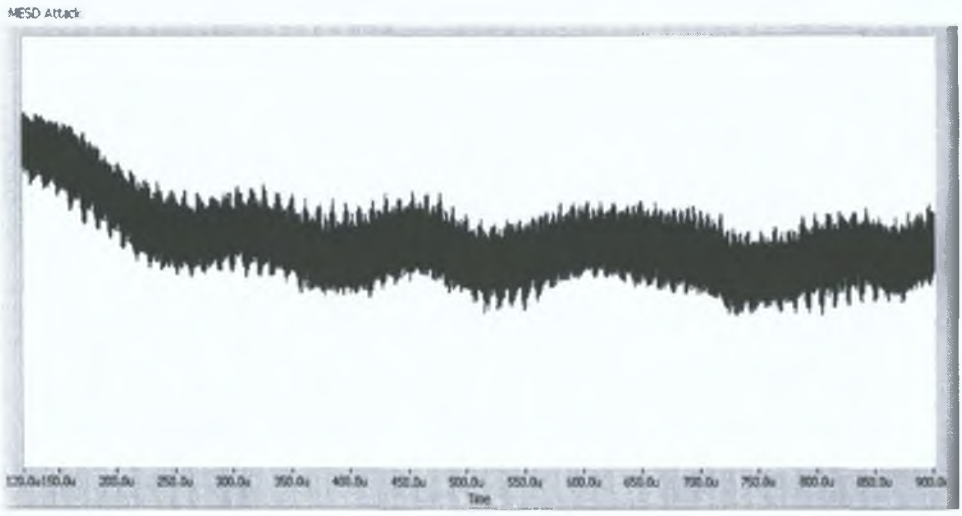


Figure 4.50: Differential trace of the two waveforms shown in figure 4.46. A difference should occur at about 580.40μs however no difference is noticed.

wasn't function correctly. It was desired to check the frequency and phase response of this amplifier however the frequency range of the network analysers available in laboratory was too high for this particular amplifier. A simple test of applying a range of frequencies from a signal generator to the input of the amplifier and measuring the amplitude of the output was carried out to give some indication of the amplifiers fidelity and no problems were observed. This method however wasn't very precise, and it gave no idea of the amplifiers phase response.

To be entirely sure that the EMC amplifier wasn't at fault, it was removed from the setup and two Picosecond 5840 broadband amplifiers were inserted instead. These offered a combined gain of 44dB well above the bandwidth of interest (up to about 10GHz) however their lower cutoff was 80kHz which blocked out the lower frequency components seen in figure 4.46. The equivalent graph of figure 4.46 is shown in figure 4.51.

Figure 4.51 was collected using the key consisting of all ones however the other waveform was identical. It can be seen here that there are a series of spikes that appear to

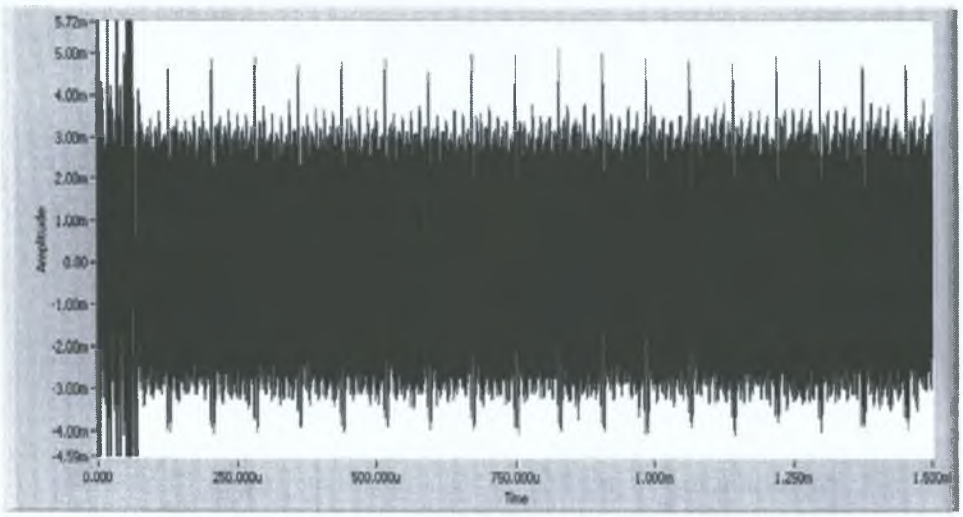


Figure 4.51: Power waveform collected over 8,000 averages. This particular waveform was passed through the two Picosecond 5840 amplifiers which filtered out the lower frequency components seen in figure 4.46.

convey secret information[21]. Unfortunately these spikes can't be explained by the little theory known about the board. They are 78μs apart which works out at 2,340 clock cycles. As mentioned earlier, the number of clock cycles required for one multiply on this particular board is 4,353 so there is no relationship between the two figures. These spikes can in fact be seen in figure 4.46 also. A magnified view is shown in figure 4.52. These spikes are clearly related to the operations on the board as they appear not to be related to the encryption operation as they fall at different point on the "humps" of the waveform in figure 4.46. They also occur for both keys and are eliminated in the differential trace. They may have something to do with some parallel operation that is occurring on the board but of course that is pure conjecture.

A close up of figure 4.51 can be seen in figures 4.53 and 4.54. Although there is information here it does not appear to coincide with the known theory of the DUT and no conclusions can be drawn on it's meaning. The DUT is such a complex piece of

[21]The cluster of spikes at the beginning of the waveform is simply the trigger signal of figure 4.37 coupling over onto the power waveform.

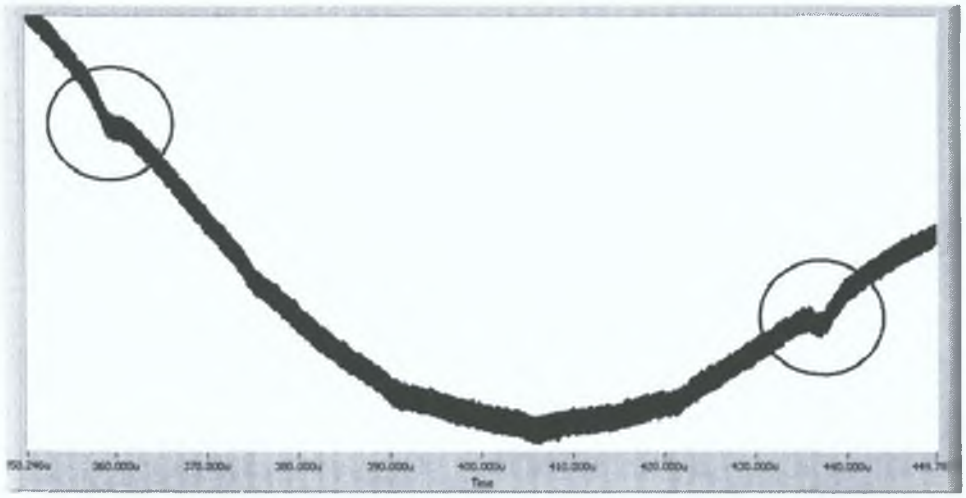


Figure 4.52: Magnified view of figure 4.46 showing anomalous 78μs spikes.

equipment it would require a great deal of knowledge about the board to relate each of these fluctuations to data operations. The fluctuation were the exact same for both keys so it is likely they were not related to the exponent.

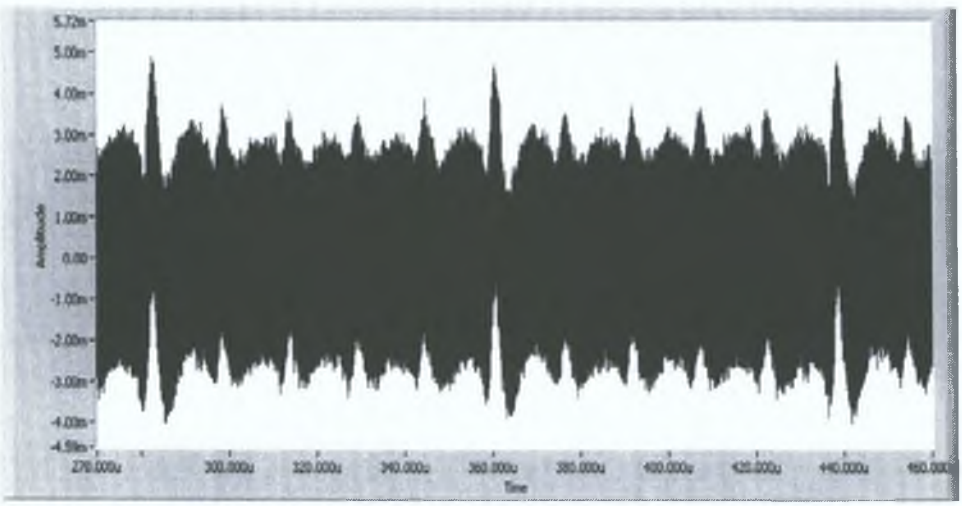


Figure 4.53: Magnified view of figure of three spikes. There is certainly a regularity amongst the fluctuation however they occur for both values of the exponent and appear not to be related to the exponent.

The final graph shown here is the one which would normally reveal the secrets to an

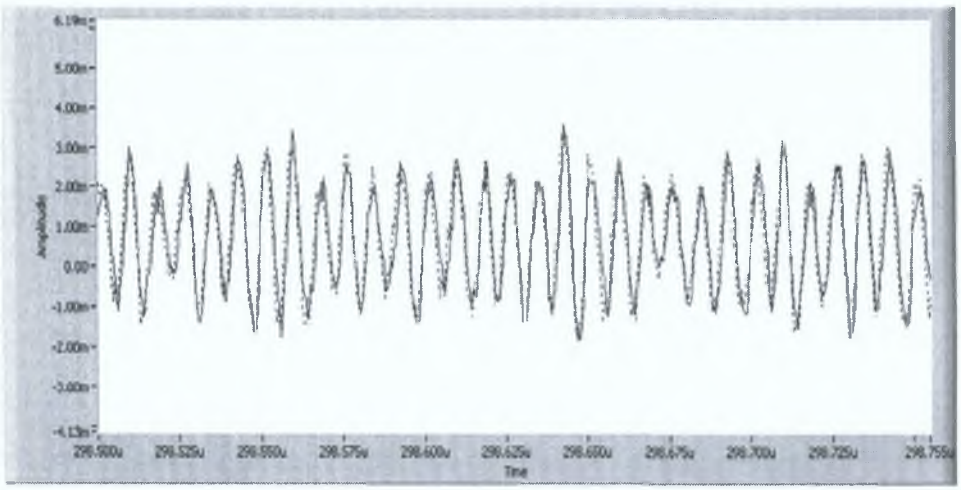


Figure 4.54: Similar view to that of figure 4.48. The waveforms for each exponent are identical again.

attacker - the differential trace. This can be seen in figure 4.55. As can be seen it does not appear to reveal any information. The reasons for this may be similar to those stated above.

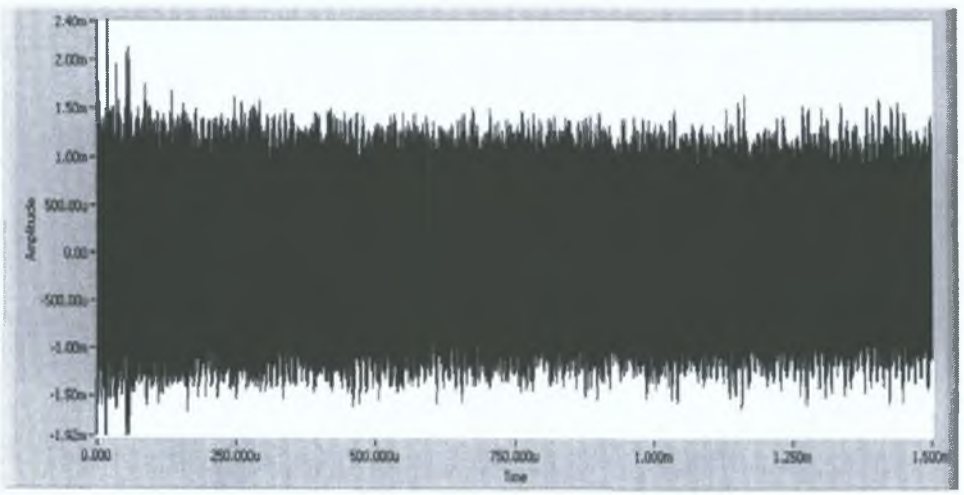


Figure 4.55: Differential trace revealing no information.

Having failed to retrieve the secret exponent using DPA it was then attempted to use EMA. The main difference here was that the antenna probes of figure 4.4 were used

in collecting the data. The advantages of this should have been twofold. If the problem lay in the Ultraview board (i.e. if it prevented the signals of interest from being observed) then this should have manifested itself as the signals were now being picked up through other means. Also by taking measurements in the vicinity of the ASIC, this should have eliminated a lot of the noise that would have been picked up using the oscilloscope probe thereby leading to shorter times for an attack. Again, the attacks failed as will be seen.

4.4 Electromagnetic Analysis

For this particular section, the setup was similar to that of figure 4.15 however this time instead of using a coaxial cable to measure the signal a number of antennas were used as mentioned. The Ultraview board was not needed at this stage and was removed from the circuit. The EMC amplifier wasn't used and the two broadband amplifiers were used instead.

4.4.1 Emulated Setup

Before attempting the actual attack it seemed prudent to run the micrcontroller system again and determine if it was possible to attack this using EMA. The results are shown in figures 4.56, 4.57 and 4.58 where it can be seen that successful results were again obtained. These results were quite significant as they showed the ease with which this data could be retrieved using and antenna and the MESD attack. Again, 2000 waveforms were collected to ensure success.

Figure 4.56 shows the data collected using the medium sized loop antenna of figure 4.4 place in a horizontal position over the center of the PIC. The data was passed through the two 5840 amplifiers to give a gain of 44dB. Two thousand averages were taken as in the power analysis section. It can be seen that the results are quite clear.

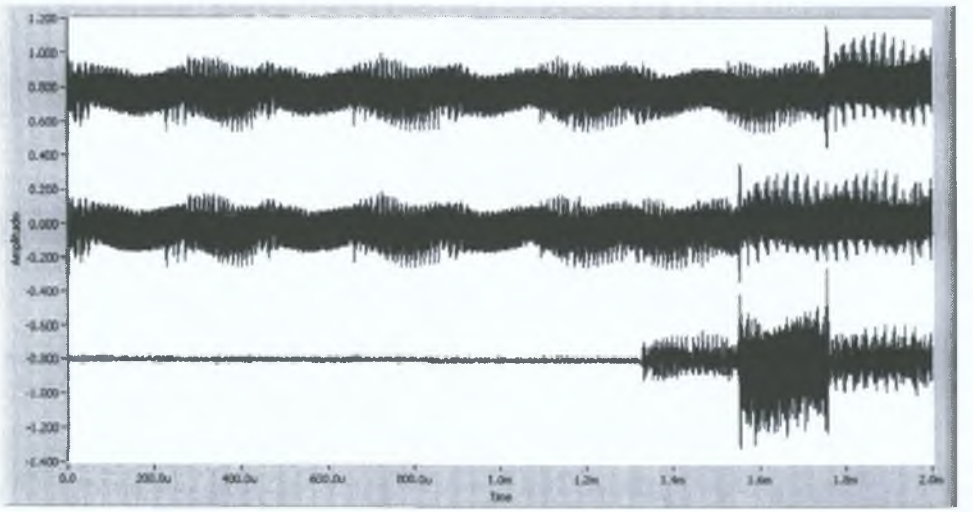


Figure 4.56: Waveform captured using the medium sized loop antenna (see figure 4.4) placed in a horizontal position over center of PIC. A similar result was produced for the monopole antenna. The value of the bit guessed at was correct as was the case in figure 4.29.

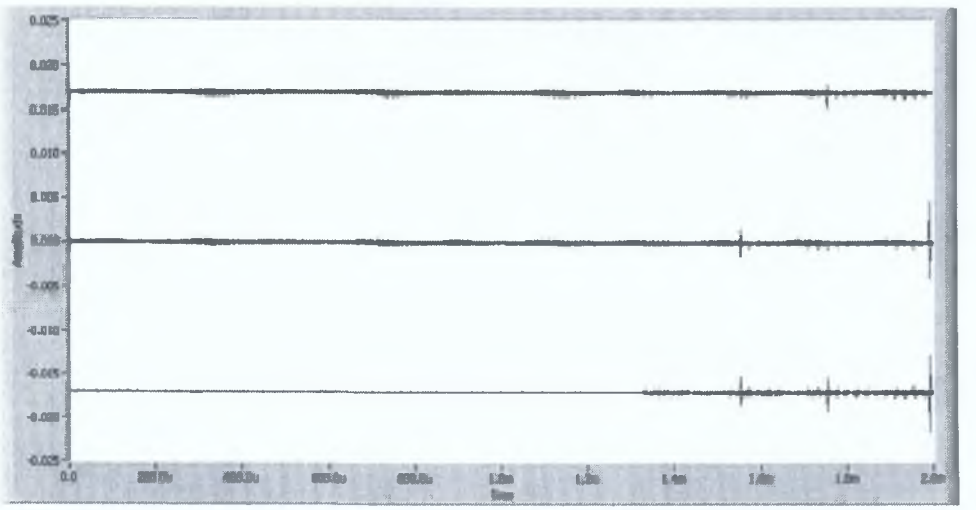


Figure 4.57: Waveform captured using medium sized loop antenna at top of PIC (near pin 1). The amplitude has dropped dramatically from that of figure 4.56 however it is still possible to determine the value of the exponent.

Figure 4.57 show the radiation captured using the same antenna however this time it was placed at the top of the PIC. The amplitude of the waveform has been reduced dramatically. The main reason for this is because the die itself is only about 5mm

square and is placed at the center of the PIC casing. Despite the reduction in amplitude figure 4.58 shows a magnified view of the differential trace which clearly allows a determination of the bit value.

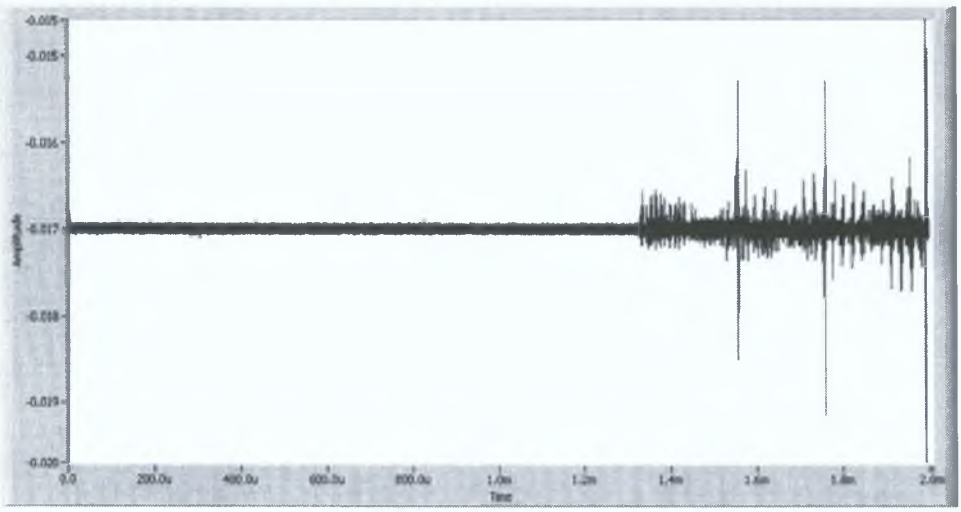


Figure 4.58: Close up of the differential trace of figure 4.57. Even placed a distance away from the source, the MESD attack was still successful.

Again the emulation verified the integrity of the setup and the next step was to perform the same steps on the DUT.

4.4.2 Basic Setup

With the experimental setup verified the DUT was again attacked. The experiment was run and 8,000 waveforms were averaged for the ball probe antenna and the large loop antenna (as this offered the greatest sensitivity to the surrounding fluctuations). It would have been desirable to take more averages however, as mentioned, the length of time to do this was highly prohibitive. The antennas were placed as close to ASIC as possible for maximum reception of the signal. Orientation of each of the antennas was very important and different signals were received for different positions.

It was necessary to be aware that the loop antennas were designed to block electric

fields an therefore only picked up the magnetic fields. The opposite was true for the monopole antenna.

Figures 4.59, 4.60 and 4.61 show the results of an EMA attack on the DUT using the ball probe antenna which picks up the electric field. Again the results are inconclusive and no secret data can be extracted from them. It can however be seen that the results for each exponent are very similar.

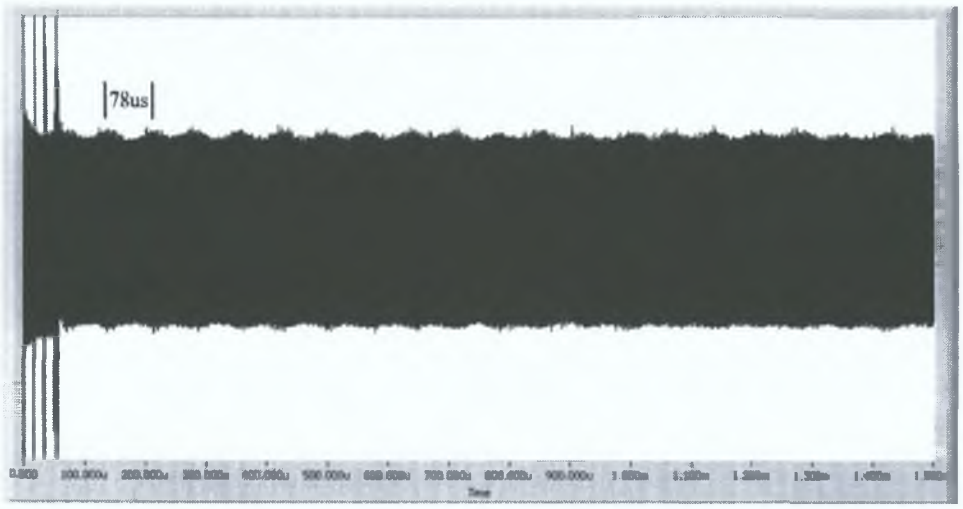


Figure 4.59: This is the waveform that was collected using the exponent consisting of all ones. The $78\mu s$ spikes can be seen as "humps" on the waveform. The waveform for the half ones, half zeros exponent is almost identical.

The differential trace for this waveform was similar to that of figure 4.55 and it revealed no information.

The next steps were to try the other antennas and to place them in different positions on the board. Each time the results were similar to that of the ball probe. The 5840 amplifiers were used in almost every case however the EMC amplifier was used for comparison on one occasion. Little differences were notices other than the amplitude. Although the EMC amplifier had a lower cut-off than the 5840 pair, the low frequency components seen in figure 4.46 were not visible due to the antenna itself acting as a

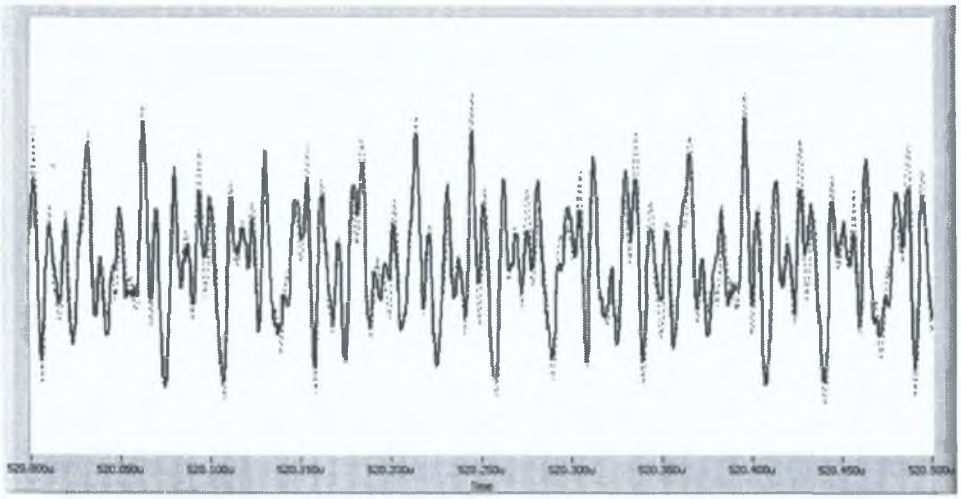


Figure 4.60: A close up of the graph of figure 4.59 is shown here. Again it can be seen that the two waveforms follow each other very closely.

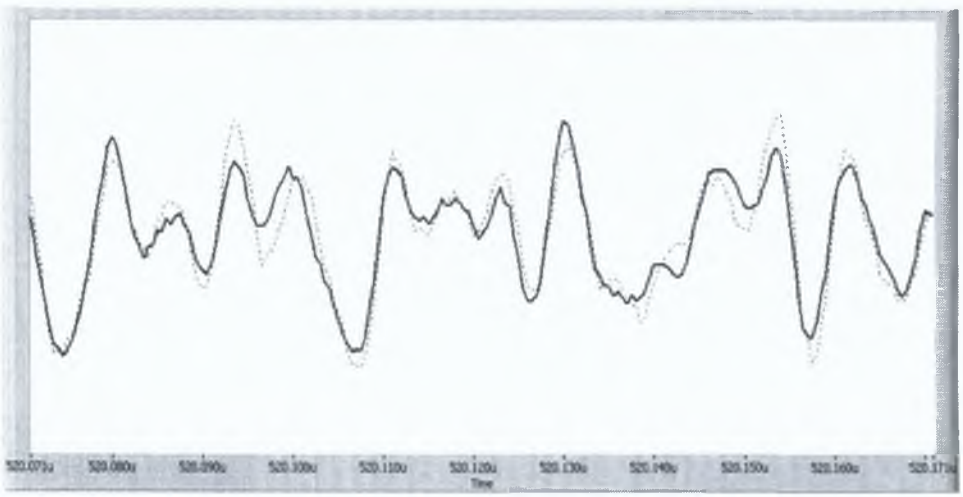


Figure 4.61: An even closer view of figure 4.59 verifying the similarity between the fluctuations. This was the case throughout the entire waveform.

filter. All in all the results proved inconclusive and the secret exponent was unable to be revealed.

4.4.3 Alternative Setup

Due to the fact that it has been shown that the data signal sometimes modulates the clock it was attempted to try an capture a harmonic of the clock an see if any information could be extracted. An 85Mhz filter and amplifier were available which was useful as the third harmonic of the clock (90MHz) fell within its bandpass. By placing these components after the 5840 amplifier stage all the other signals would have been filtered out and the hope was that only the 90MHz harmonic and it sidebands would be passed.

When using the 85MHz filter and amp, the spectrum analyser was initially used to demodulate the signal received. The sound was turned on to allow audio inspection of the resulting demodulated signal. As the antenna was brought closer to the cryptographic processor, the audio signal increased in amplitude suggesting that a modulated signal was present. Unfortunately the MESD attack again produced not results.

It was then attempted to bypass the spectrum analyser and take the signal directly from the 85MHz amplifier. Although the signal wasn't demodulated this shouldn't have really made a difference as the power attack has an inherent demodulation scheme built into it. As was now expected this produced not results.

As one final attempt at and attack, a yagi antenna was obtained. This would allow measurements of the radiated emissions in the far field (which are true radiated fields and not reactive fields). However it was realised that the most success would have been achieved if the higher harmonics were attacked. The problem was that only the 85MHz amplifier and filter were available. To allow measurement of the higher harmonics a mixer and signal generator were obtained which would down convert this signal to 85MHz. The setup was essentially a superheterodyne structure. Although this appeared to work well initially repeated attempts at extracting the secret exponent failed.

A picture of the general setup used for these experiment is shown in figure 4.62. Although it is not optimised for noise reduction this shouldn't have caused too much of a problem. Of course to improve the attack certain noise reduction measures could be taken such as the use of a shielded enclosure.

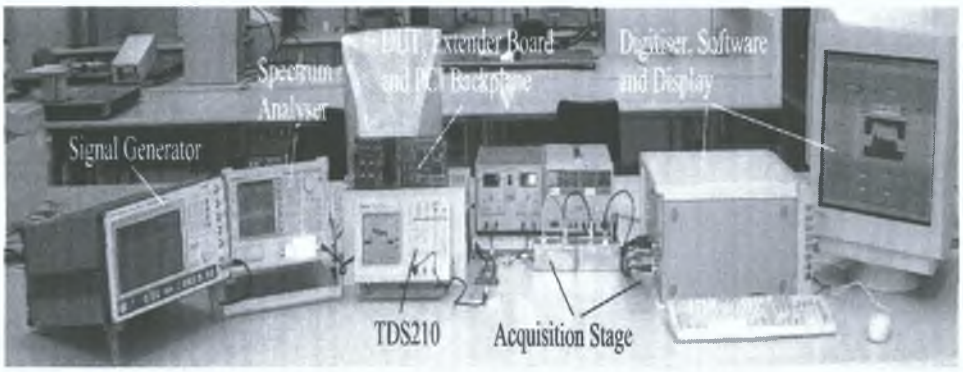


Figure 4.62: A picture of the general setup for the DPA attacks outlined in this chapter.

4.4.4 Other strategies attempted

As the MESD attack didn't work against the DUT in it's basic form, it was attempted to improve the analysis using some different methods described in the literature. One of these was similar to a total power analysis (discussed in the next chapter). This involves calculating the total power in a windowed section of the waveform. The idea was that if the register update caused an increase in supplied power then this would cause the total power of the particular windowed section of the waveform in which a 1 is operated upon to be different from that in which a 0 is operated upon. The time window chosen was the length of time it took for one multiplication (i.e. $145.1\mu s$). The window was then "slid" across the waveform and the total power calculated at each point. A number of peaks and troughs were noticed in the resulting waveform but nothing of significance.

Cross-correlation was then used as a second attempt. This involved taking a portion of

the wave which represented on multiply and cross-correlating this with the averaged waveforms for each key. It was hoped that the resulting waveforms would be different in some way however this was not the case and the two were identical.

A spectral analysis was performed on the two averaged waveforms for each attack attempted and the two spectra compared. Each time little or no differences could be noticed.

A higher order differential power attack was then attempted. This involved taking all the signals collected and performing a joint MESD attack on them. To accomplish this two sets were created. The first set contained the waveforms representing the key with all ones whereas the second set were those with half one half zeros. Each set was averaged and a differential trace obtained. Again the results were inconclusive.

4.4.5 Summary

This chapter looked at the actual experimental setup and results. It was seen that the attacks worked both in simulation and on a PIC microcontroller emulation however they failed in every respect on the DUT. This would suggest a level of security on the board. Some possible reasons for this failure were given. Due to the fact that one of these reasons was the equipment, it should be kept in mind that it may very well be possible for a better equipped attacker to break this device. Because of this possibility, and the fact that these attacks are clearly a real threat to other new designs, some potential countermeasure are given next.

Chapter 5

Countermeasures

It was shown in the last chapter that numerous attempts at attacking the DUT failed even though success was achieved with a microcontroller running a similar algorithm. Despite this, it is not safe to draw the conclusion that the DUT is completely secure. It is generally accepted that no current, practical cryptographic algorithm (or system) is 100% secure, and constant vigilance on the part of the designers of cryptographic systems is necessary.

Whether or not an algorithm or system is completely secure is not something that can easily be proven and in most cases never has been. The cryptographic community will tend to accept algorithms that withstand years of attacks (such as RSA) even though nobody has ever proven the security of these algorithms. However, unless a solid proof exists, it is never safe to say that an algorithm or system is 100% secure. It is therefore prudent to assume that an adversary with better funding and more advanced equipment would be able to successfully attack this board. Due to the high probability of this scenario, it is necessary to consider some of the potential countermeasures that have been proposed in the literature.

Although each countermeasure may offer a fairly substantial reduction in the compromising emissions, none are perfect and no one technique counteracts all possible attacks. However, a well thought out combination of these and other mitigation techniques should provide an increased level of protection[1].

[1] Of course the cost of this should not exceed the value of the information being protected.

Any design engineer will agree that an emission problem should be resolved before it appears - this is standard teaching in the field of EMC. It is a lot more costly to fix an emission problem at the end of the design stage than at the beginning. It is therefore imperative that emission control (be they power, time, radiated etc.) be made part of the overall design process[2].

Although a lot of the approaches proposed to date have been aimed mainly at smart-cards, the same techniques can be applied to cryptographic accelerators. In fact, it should be considerably easier in some cases to implement the same (and even improved) strategies on these devices as they have greater flexibility in their size, speed and memory capacity.

The following chapter is split into two main sections. The first section deals with power analysis countermeasures and the second deals with EMA countermeasures. Despite this fact, a lot of the countermeasure are effective against both attacks.

5.1 Countermeasures against Power Analysis

According to Messerges et al. [76], software countermeasures are among the more practical techniques used to protect systems. The reason is mainly due to it's flexibility which ultimately leads to low cost. Software countermeasures are also proposed by Kuhn et al. [66] to protect against TEMPEST attacks on VDUs as will be seen. Despite this however, Clavier et al. state that in order to reduce the feasibility of side channel attacks both hardware and software countermeasures should be used together [88].

Chari et al. report that most of the attacks proposed were only ad-hoc and did not constitute secure countermeasures [89]. The approach taken here is more of a firm mathematical approach where a model of the cryptosystem and it's side-channels is

[2]It is clearly not possible to envisage every problem that may occur but at least if a thorough approach is taken at the design stage all subsequent problems should be minimised.

produced and lower bounds are identified. The bounds show the minimum amount of samples that need to be taken in order for an attack to be successful.

5.1.1 Timing Randomisation

One of the methods proposed is that of *timing randomisation*. This involves placing random time delays into the software so that a power analysis will not be possible. It was mentioned earlier that a steady trigger is vital if a power attack is to be successful. This is due to the fact that the sample points must line up with each other if averaging is to be effective. With random delays introduced, a steady trigger will not be sufficient to allow the averaging to work and will therefore act as a countermeasure. Each time window identifying the secret parameter is effectively shifted to different points on the power waveform for each trace wf_{jk}, thus reducing the effect of averaging. If the number of possible delays is d and each delay is equally likely then the resulting bias signal will not have a size of ϵ as before but will in fact only be ϵ/d and there will be d bias signals instead of one.

Methods exist that can defeat timing randomisation and still allow the power attacks to be successful. Some authors report that randomisation can be defeated by signal processing techniques [35, 89] however Messerges claims that although this is likely to be the case, timing randomisation is a powerful tool and with the right development could probably withstand any signal processing techniques. Despite this, the present versions are "naive" implementations and can be defeated quite easily.

One attack on timing randomisation is proposed in [76] and is known as *total power analysis* which performs the following operation on the data:

$$P_{total} = \sum_{j=0}^{w-1} (\Delta d_k)^2 \tag{5.1}$$

where P_{total} is the total power of the window w and Δd_k is the kth sample of the differential power waveform. In the general case the waveform with the correct guess will show a greater total power to the one with incorrect guesses.

Another method of attacking this method is known as a *correlation attack*. In this attack a chosen template of the power signal is used to search through the randomised waveform and to pinpoint the locations of the sections of code that differ for each key. Once the locations are known, the waveform can be arranged so that similar sections align up together for correct averaging. More detail can be found in [76].

5.1.2 Random Process Interrupts

A similar idea to timing randomisation is that of *Random Process Interrupts* (RPIs). This basically consists of inserting "dummy instructions" between the actual instructions of the processor. As with timing randomisation this reduces the effect of averaging due to the fact that the waveforms won't be aligned correctly at all sample points. RPIs are a more general case of timing randomisation as the instruction can be an active instruction and not just a delay. An active instruction will tend to have it's own power fluctuations and will therefore introduce increased levels of noise.

The difference between RPIs and timing randomisation that Messerges describes is that the timing delay is performed every round with equal probability for 0 to d-1

Performing a differential power attack on a device without any countermeasure should produce a spike with amplitude ϵ (as discussed in chapter 2). In order for this spike to be visible it must be larger than the noise that surrounds it. Therefore,

$$\epsilon > \frac{\sigma}{\sqrt{N}} \tag{5.2}$$

where ϵ is the mean value of the spike, σ is the standard deviation of the noise and N

is the number of power waveforms collected in the DPA attack.

Using RPIs on a particular algorithm will have the effect of spreading the bias signal over k clock cycles each one being some fraction of ϵ and the worst case value for the largest spike being ϵ/k. The value k is determined probabilistically. The occurrence of this spike is distributed normally over the waveform and can be characterised by a mean value μ_{rpi} and a standard deviation σ_{rpi}. As the standard deviation is the "spread" about the mean this implies that the bias spikes will be spread over twice this value. Clavier et al. show that

$$\begin{aligned} \sigma_{rpi} &\approx \sqrt{np} \\ \Rightarrow k = 2\sigma_{rpi} &\approx 2\sqrt{np} \end{aligned} \tag{5.3}$$

where k is the number of clock cycles the bias spike is spread over, n is the number of clock cycles after which it would have appeared if no RPIs had been included and p is the probability that an RPI occurs in any particular clock cycle.

As a result of the spreading, the bias spike will be reduced in size. In order to identify the largest spike, as before, the following should hold

$$\frac{\epsilon}{k} > \frac{\sigma}{\sqrt{N'}} \tag{5.4}$$

where N' is the number of power waveforms taken in this case.

To keep the signal to noise ratios the same for both cases the number of waveforms needed are

$$\begin{aligned} \frac{\epsilon}{\sigma/\sqrt{N}} &= \frac{\epsilon/k}{\sigma/\sqrt{N'}} \\ \Rightarrow N' &= k^2 N \end{aligned} \tag{5.5}$$

In other words the attack will be just as successful if k^2N power signals are collected. However as Clavier et al. show it is not really feasible to collect these. Assuming that $p = 12\%$ and $n = 1600$ implies that $k^2 = 768$. Assuming also a best case scenario of $N = 100$ this implies that $N' = k^2N = 76,800$ which make the attack a lot less feasible.

As with timing randomisation, the RPIs can sometimes be removed with signal processing techniques which would reduce N' down to a more reasonable value. Two methods which can be used to remove them are also shown in [88].

5.1.3 Internal power supplies

Power supply filtering would be another method that could be used to reduce the possibility of a power attack. For example, Adi Shamir proposes building a simple capacitance network into each smartcard to allow the fluctuations to be contained within the smartcard itself and thereby preventing power attacks [90]. The proposed architecture is shown in figure 5.1 were it can be seen that two capacitors and a four extra diodes are needed. The cost of each of these would only cost a few cents extra per smartcard according to Shamir. The problem with this method is that although it would reduce the possibility of a power attack, an EMA attack would still be feasible as the fluctuations themselves haven't been eliminated.

Rakers et al. describe a contactless smartcard which they claim is protected against DPA. This is due to the fact that it uses an isolation circuit that contains a current source with a value "completely independent of the activity or power requirements of the digital circuitry". The circuit has a value of 66dB of isolation which increases the required time for a DPA attack by a factor of 2^{22}. This is a similar idea to Shamir's although it is slightly more sophisticated. More information can be found in [91].

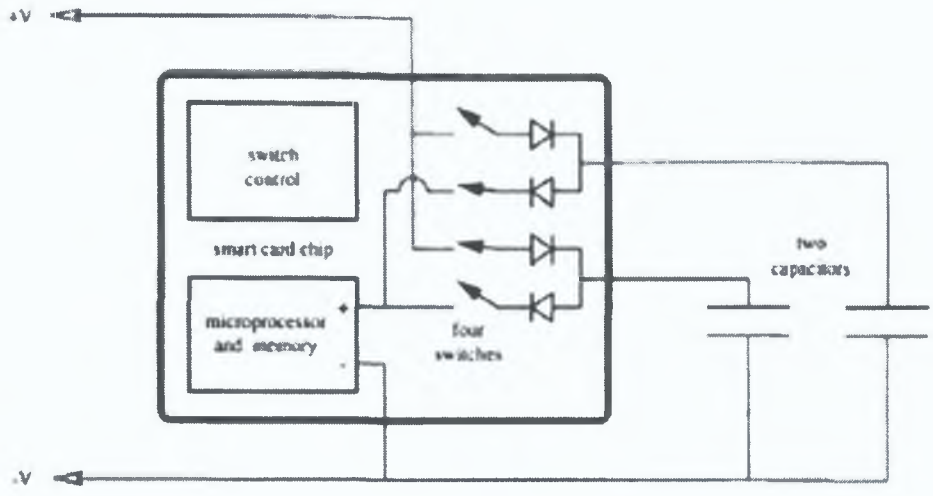


Figure 5.1: Detached Power supply proposed by Shamir.

5.1.4 Data masking

One of the methods proposed in [89] and [76] consists of *masking* the intermediate data (i.e. mask the input data and key before executing the algorithm). This would make the power fluctuations independent of the actual data. Two types of masking include *boolean masking* and *Arithmetic masking*.

Boolean masking using the XOR operator to mask the intermediate data. For example, consider a word x that is required to be masked. Using boolean masking and a random mask r_x the masked value is

$$x' = x \oplus r_x. \tag{5.6}$$

Arithmetic masking on the other hand uses addition and subtraction modulo 2^w where w is the word size being operated in. Using the same values as in the last example an arithmetic mask would be

$$x' = (x - r_x) \bmod 2^w. \tag{5.7}$$

Unfortunately however Coron and Goubin pointed out that although this method works

well for the classical algorithms such as DES and RSA, the case is not so clear cut for algorithms that combine boolean and arithmetic functions. These algorithms need to use both boolean and arithmetic type masking and therefore require a method switching between the two types. The methods proposed by Messerges for doing this is apparently not sufficient to prevent a DPA attack. More details can be found in [92] and a secure approach can be found in [93].

5.1.5 Tamper Resistance

Tamper resistance was briefly discussed in chapter 1 and seems to be a viable countermeasure. However, as mentioned previously, it is unlikely that tamper resistance would protect against an EMA attack due to the fact that it is almost impossible to detect an the attackers antenna[3].

The perfect tamper resistant device seems like an elusive goal as a cunning attacker will always find a new and better way to surpass the tamper resistance mechanism. Anderson et al. give a cautionary note in [16].

5.1.6 Fail Counters

A differential power analysis attack requires the attacker to obtain a significant number of power waveforms. In order to do this the attacker must have the ability to run quite a few encryptions on the system under attack. If the number of encryptions were limited to a certain number then the attacks would become increasingly difficult.

This solution may have it's own problems also as it might be necessary to perform quite a few encryptions legitimately. Also if the attacker can manage to reduce noise to a very low level then the number of waveforms required may be reduced below the fail

[3] If the antenna were to measure the radiation in the near field then it might be feasible to detect its presence however this could simply be the chassis of the host system absorbing some of the electromagnetic energy.

count number. One way of significantly reducing the number of waveforms required is to use *synchronous sampling* as described in [76].

5.1.7 Removal of conditional elements

One of the main features used to attack the square and multiply algorithm is the fact the it has a conditional multiplication that depends on the value of the exponent bit being operated upon. One suggested countermeasure is to implement this multiplication in every round (regardless of the value of the bit) and to only do a register update when the bit is a 1. Unfortunately as was seen in section 4.3.4 the MESD attack doesn't rely solely on the conditional multiplication. However it does increase the number of waveforms required for an attack.

An improved approach was also tested in which the multiplication was performed every round along with two register updates - one being a dummy register. This increased the number of waveforms required even more than simply performing the multiplication on every round.

5.2 Countermeasures against EMA attacks

The Idea of EMC is of a similar nature to the TEMPEST problem and a lot of useful information can be taken from the studies of EMC. It is highly recommended that designers wishing to develop a device secure from the threat of electromagnetic eavesdropping be fully familiar with the techniques used in EMC. However, it must also be stated that a device which is fully compliant with EMC standards will generally still emit enough emissions to allow adversaries to reconstruct secret information. In [60] it is shown that even though the VDUs all complied with the CISPR standard, it was still possible to pick up and reconstruct the video signal.

5.2.1 Asynchronism

Asynchronism seems to be a viable method of mitigation. As mentioned in the experimental section one of the problems that was encountered was the inability to trigger off the waveform at the same point in time for every acquisition. This reduced the effectiveness of the averaging. Once a valid trigger was established however the results improved dramatically.

Not only would asynchronism stop the attacker from gaining a valuable trigger, it would also eliminate the need for the clock. As discussed in chapter 3, due to its high frequency components, the clock is the cause of some of the strongest radiated signals in digital systems[4]. If the clock gets modulated by the information being processed, then this data can be effectively radiated from the board - some interesting results to this effect are given in [32].

By eliminating the clock altogether not only does it eliminate any triggering information but it also eliminates any potential carriers the secret data may try to "piggyback" off. As discussed in [72], synchronism causes the operations to occur at the same time which increases the radiated emissions. However if the operations occur asynchronously to each other, then the radiated emissions can be almost 30dB down from those radiating from a synchronised device.

Incidentally, another method of reducing the radiated level of the clock is to apply a technique known as *spread-spectrum modulation*. Spread-spectrum modulation is a modulation technique generally used to increase the security of a communication system. The transmitted data is "spread" over a bandwidth that is in excess of the minimum bandwidth needed to send it. This is accomplished using a code to modulate the original data. The receiver needs to have the same code to "despread" the received

[4]However the switching current of each of the transistors would still exist and this would produce high frequency radiating components. However in this case the spectral content would be spread over a larger bandwidth decreasing it's effectiveness.

signal. Using this technique on the clock signal of a system, the power of the radiated signal is spread over a larger bandwidth resulting in a reduced emission at any particular frequency. This is described in more detail in [94].

5.2.2 Shielding

In order to protect a device from external sources of radiation, and to stop radiation from leaving a particular area, the device needs to be shielded. Shielding is one of the universally accepted ways of reducing unwanted radiated emissions and basically involves placing a sheet of conductive material around the device to block the radiation. This idea is by no means a new one and is an obvious method to look at in order to stop sensitive information being compromised. In military situations where a system is not processing "classified" information, the level of shielding need only be good enough for EMC (Electromagnetic Compatibility) standards, however "TEMPEST shielding" is required if classified information is being processed and is naturally more complex than that used for EMC purposes [95, 96].

Shielding that blocks electric fields does not necessarily block magnetic fields. It may therefore be required that ferromagnetic material (such as mu metal) be used in conjunction with the other type. However, as stated in [72], different information is contained in each field[5] so testing should show which is more vulnerable to attack and then shielding can be applied more heavily for that case.

Shielding is considered more of a remedy than a cure and should really only be relied upon in the case where redesign is a costly operation. As stated on a number of occasions, it is better to minimise the radiated emissions during the design stage rather than relying on methods such a shielding to contain the radiated emissions. However, it will sometimes be the case that not every problem will be foreseen so early in the

[5]This only applies in the near field of course as the only difference between the electric and magnetic field in the far field is a factor of 120π.

development cycle.

Although it is not a requirement for TEMPEST approved equipment to suppress radiated emissions of non-sensitive signals, it is recommended in [59] that all emissions be reduced as much as possible. The main reason for this is the cost of testing. Although the signals may very well not contain sensitive information, they must be tested against and as a result the time and cost of the testing increases.

The first and most obvious method for counteracting TEMPEST attacks is to provide adequate shielding. All of the attacks outlined above are clearly undesirable and if a cryptosystem is to survive in the marketplace it must be able to withstand these attacks. The methods used at present in commercially available sources more than likely do not fully protect against all of these attacks and there needs to be some new protections devised. Some of the features of a popular commercially available device are discussed in the next chapter and these should be a starting place of the development of these mitigation techniques. However, to begin with, it will prove informative to discuss some of the techniques described in the different papers and documents referred to in the bibliography. For TEMPEST emissions it is clearly necessary that efficient shielding is in place to block the radiation containing the sensitive information. For power analysis, timing attacks and fault induction attacks other techniques will need to be used also.

Although it is shown in [60] that it is possible to reconstruct signals at a distance of 50 metres from and unshielded VDU, this value is dropped to only 10m with the shield in place. This is still a significant distance in terms of security.

With adequate shielding in place a low pass filter should then be placed at the power line in order to stop any high frequency coupled information from passing outside the module. It will be seen in the next section that this is the method the IBM4758 (the

first cryptographic processor to gain FIPS 140-1 level 4 security rating) uses.

5.2.3 Balancing

A lot of the techniques proposed here are a lot more feasible on cryptographic systems (such as the one under test here) than on smartcard systems because of the extra space available. *Balancing* is one such technique. This involves negating the effects of one set of events by performing a complementary set [97]. This technique is unfortunately very difficult to implement correctly.

5.2.4 Red/Black Separation

It is important not only to minimise radiated emissions from a device but also to minimise conducted emissions. If a section of a system is adequately protected against radiated emissions, then it will not be possible for an attacker to receive information with an antenna placed in that area. However if the attacker moves the antenna to another location (over the power line entering the system for example), then it is very possible that the protected signals will be coupled onto the power line and transferred outside the system. The power line being a fairly large cable will act as an efficient radiator of electromagnetic energy and may cause even more of a problem than the original radiated signal. So it is necessary to consider conducted emissions also. However, if subsystems are separated from each other using the RED/BLACK guideline described in [51] then this may not be as much of a problem. This guideline basically states that systems operating on sensitive data (RED systems) should be physically isolated form those operating on non-sensitive data (BLACK systems). As well as this, this every system should be carefully designed so as reduce emissions and some techniques for doing this can be found in [98].

5.2.5 Provably secure Countermeasures

Most of the countermeasures proposed to date will only work under certain conditions. For example it might be assumed that the attacker can only collect a certain number of waveforms or that access to a certain piece of equipment will not be available. In general it is dangerous to make such assumptions and a worst case scenario should be considered for maximum security. This of course is not always economically feasible and a balance should be met.

Generally it will not be known for certain whether or not the device is completely secure. However an ideal situation might be to develop a mathematical model of a system and use this model to design *provably* secure countermeasures. This was proposed by Chari et al. in [89] where it states

> "A scientific approach is to create a model for the physical characteristics of the device, and then design implementations provably secure in that model, i.e., they resist generic attacks with an a priori bound on the number of experiments."

The model would of course have to be simplified greatly which in itself would increase the possibility of failure in the countermeasures. This is seen in [97] where Quisquater shows that Chari et al.'s first approximation which ignores the coupling effects between systems in close proximity to one another, will not work. This is due to the fact that EMA wasn't known at the time [89] was written. Despite this, a provably secure countermeasure is a very desirable result.

5.2.6 Summary

This chapter looked at some of the mechanisms by which the attacks discussed in this thesis may be protected against. Unfortunately no one technique will suffice and a well

thought out combination of them will generally be needed. Some of the countermeasures overlap for the two attacks discussed. For example, masking the intermediate data operations will not only reduce the possibility of a power attack but will reduce the possibility of an EMA attack also.

The techniques used to protect cryptographic systems from these attacks should not cost more than the information being protected. A balance between the level of security and it's cost should therefore be found. To decrease the costs, all problems should be tested for at the design stage when it is a lot less easier to eliminate them.

There are other proposed techniques available that haven't been described here and which can be found in [99–107].

Chapter 6

Conclusions

Side channel attacks have been shown to be successful in compromising the secret information on smartcard implementations of cryptographic systems. These attacks take advantage of some characteristic of the physical implementation of a cryptographic algorithm and are not only a threat to smartcards but also to more complex cryptographic systems.

This thesis attempted to determine the level of vulnerability of a recently developed high speed cryptographic accelerator to these side channel attacks. Using the obtained results, the designers will be able to determine with increased confidence, the extent to which the device conforms to cryptographic standards that require mitigation techniques.

Although a number of side channels exist, only two were examined and implemented. This was due to the fact that they posed the greatest threat to the particular system under review. The two attacks are known as power analysis and electromagnetic analysis.

Power analysis allows the attacker to gain a knowledge of the secret parameters residing on a cryptosystem. It accomplishes this by measuring the fluctuations in the power the systems draws from its supply. In a similar manner, electromagnetic analysis allows acquisition of this knowledge by capturing and analysing the electromagnetic radiation emanating from the system. The US government has been aware of these attacks for quite some time and has set up a classified program known as TEMPEST to

protect against them. Although TEMPEST is thought mainly to deal with electromagnetic emissions, it is likely that it deals with other forms of compromising emissions also, including power attacks.

Power analysis was studied first as there is a firmer understanding of this attack in the public domain. It was seen that the power drawn by cryptographic systems is data dependent. The power drawn when a 1 is being manipulated is slightly different from that required when a 0 is being manipulated. By using statistical techniques, it is sometimes possible to tell the values of individual bits. Averaging was seen to be the main cause of success for power analysis. By running the algorithm thousands of times the noise is reduced to an insignificant level, however the fluctuations caused by the operations on the board remain. The specific values of certain bits can be determined by calculating the differential trace which essentially "demodulates" the required information from useless periodic information not eliminated by the averaging.

A logical progression from power analysis is Electromagnetic Analysis (EMA). The main difference between the two is the method used for collection of the signals, however there are subtle differences such as modulation effects and localisation properties of EMA that are not of concern in power analysis. These effects can, in some instances, make EMA a more powerful attack than power analysis. The theoretical chapter on EMA presented a number of models that can be used to predict the radiation for electronic circuits. This then led to a discussion of the TEMPEST program however due to it's classified nature, little practical insight was gained. Finally, a literary review was given and it was learned that the realisation that electromagnetic radiation could be used to compromise electronic systems is fairly new. Researchers have discovered that indeed the attacks are a real threat.

Following the theory, the next chapter presented the experimental results and procedures. Experimental techniques were set up to attack the DUT. The techniques were

tested with both a simulation and a microprocessor emulation and then used to attack the DUT. The following results were observed:

- The simulation produced waveforms that resembled those of a cryptographic processor and included simulated bias spikes and noise (both periodic and random). These waveforms were then "attacked" using the Labview programs written for the main attack. The attacks were successful and the bias spikes were revealed.

- As a more realistic test (and one which tested the whole setup as opposed to just the software), a microcontroller circuit was designed and built. This circuit emulated the operations of the DUT by performing encryption using Montgomery's method for modular exponentiation. The technique was different from that used by the DUT but was sufficient to test the setup. The attack was done using both an ordinary MESD attack and an EMA attack. The attack was successful in both cases and revealed the value of the secret exponent. Some countermeasures were attempted in this case and were found to be helpful in increasing the level of difficulty of the attack. It was quite a significant result that the microcontroller circuit was compromised with ease using the MESD attack and an antenna, as it would be very difficult to detect this form of attack. It thus poses a significant risk and it is necessary that action be taken when designing new systems.

- With verification that the entire setup was sound and that a real threat existed, an attack was attempted against the DUT. A spectrum analysis was performed using a spectrum analyser, and Matlab software was used to analyse the results. No information about the exponent was revealed. A power analysis was then attempted using a number of different probes. Although the operations of the board were clear, no information about the secret exponent was revealed.

- Finally an electromagnetic analysis was attempted with a number of different antennas at a number of different frequencies. A super-heterodyne architecture was used to down-convert the higher frequencies to the intermediate frequency of the available amplifier. Although these attacks were successful against the microcontroller circuit they failed to produce results against the DUT. This only served to increase the level of confidence in its ability to withstand these attacks.

In all cases the attacks against the DUT failed. These included

- The initial comparison of the spectrum for each of the key,
- The powerful MESD attack using measuring probes which had been successful in the simulation and the emulation,
- The EMA attack using the magnetic field antennas, the electric field antennas and the Yagi antenna,
- Some additional techniques such as total power analysis, cross-correlation and high-order differential power analysis.

There are a number of possibilities for the failure of the attacks to reveal the exponent of the DUT. Some of these include

- The sampling rate of the digitiser was very prohibitive. A maximum real time sample rate of $100MSs^{-1}$ was very prohibitive for a system that runs at 60MHz (and even 30MHz).
- The analog bandwidth was only 100MHz. This would have filtered out many of the higher harmonics.
- The resolution of the board may have been a problem. The bias spikes may have been lost in the quantisation noise caused by the 8 bit resolution.

- The Ultraview board may have been filtering the signal of interest at the measurement points. However, the DUT itself must have been receiving the correct power otherwise it would have failed to operate. The fact that EMA failed also seems to suggest that the Ultraview board was not the problem as it wasn't used in these later attacks.

- The design of the DUT itself may very well have been the cause for failure. The fact that the multiplication of the square and multiply algorithm was done every round would certainly increase the number of waveforms required for an attack. This was one of the countermeasures suggested in the literature and may have made the 8,000 waveforms insufficient to reveal the secret key. Also, the parallelism of the board may have been a major contributor. The board has 10 ALUs all running parallel to each other. This would certainly be a major contributor to the failure of the attack.

Despite the fact the DUT passed all the tests it is not safe to draw the conclusion that it is secure. It is very possible that the equipment was the problem and this must be kept in mind. The MESD attack was shown to be a very powerful attack and this is a reason why the DUT is considered vulnerable to an adversary with improved resources.

It was seen that although the ultimate goal of the attacks failed, a lot of information about the device was learned in the process. This can also hold serious consequences as it may indirectly lead to knowledge about the secret key. Reduction of events that lead to any information must therefore be a major consideration in the design of new systems.

Although the main attacks failed, it is clear that a problem does exist. This was not only shown by referring to numerous published papers on the subject but also by experimental results. Designers of cryptosystems need to keep this in mind. By following

the methods of this thesis it will be possible for an engineer who is completely new to this area to devise tests for a cryptographic device. Other side channel attacks must also be kept in mind. These include timing attacks, fault induction attacks and optical TEMPEST [108, 109].

Some improvements for future testing and research might include the following:

- Acquisition of a high quality digitiser. This would preferably mean one that samples at a minimum real time sample rate of 20 - 30 times the clock speed of the board under test. Its analog bandwidth should be about 10 times[1] this clock rate to ensure signal integrity. The digitiser should also have a resolution of at least 12 bits. Unfortunately, the higher the clock rate of the board the more expensive the required digitiser becomes, however for the 60MHz DUT, a suitable digitiser is available from Gage Applied inc. [80].

- For power analysis, measuring the power on the power pin of the ASIC would be highly advantageous as it would eliminate a lot of noise. Although this might be more difficult for an attacker than a designer (due to possible tamper resistant measures and the fact that the pin would have to be temporarily disconnected from the PCB board to take the measurement), it should not be dismissed as a possible point of attack.

- Obtain active[2], high frequency measuring probes and high quality, low noise amplifiers and filters of the appropriate bandwidth.

- For an EMA attack, a high quality, broadband receiver that allows measurement of the demodulated signal would be an advantage. This would replace the "home made" superheterodyne receiver used here.

[1] It is possible that a lower bandwidth will still produce positive results also.
[2] As opposed to the passive probes used for measurement here.

- High quality receiving antennas of the appropriate bandwidth would also improve results. An array of coil antennas (such as the one described in chapter 3) could be used and placed strategically over the entire area of the ASIC. This would increase the possibility of locating the desired signal.

- Increase the number of waveforms collected. The number collected for this experiment was 8,000 for each section. If this was increased to about twice this number then there is a greater chance of the attack being successful.

The improvements given here are of course only a few possible suggestions and there are many other things that can be done to increase the possibility of a successful attack. It is necessary for a designer to try to anticipate every possible way an attacker can improve these results if a secure cryptosystem is to be obtained.

Finally, it is clear that this research is only a small part in the ongoing struggle to obtain the perfect cryptosystem. However, by learning from the problems encountered in this thesis and the solutions to resolve these problems, designers can be that bit closer to its implementation.

Bibliography

[1] Federal Information Processing Standards Publications (FIPS PUBS) homepage. Available from http://www.itl.nist.gov/fipspubs [Accessed 14 May 2003].

[2] B. Schneier, *Applied Cryptography*. Wiley, 2nd ed., 1996. ISBN 0-471-11709-9.

[3] National Security Agency Website. Available from http://www.nsa.gov [Accessed 12 May 2003].

[4] E. Biham and A. Shamir, "Differential cryptanalysis of DES-like cryptosystems," in *Advances in Cryptology - CRYPTO '90 Proceedings*, pp. 2–21, Springer-Verlag, 1991.

[5] J. Daemen and V. Rijmen, *The Design of Rijndael: AES - The Advanced Encryption Standard*. Springer Verlag, 1st ed., 2002.

[6] W. Diffie and M. Hellman, "New directions in cryptography," *IEEE Transactions on Information Theory*, vol. IT-22, pp. 644–654, November 1976.

[7] W. Stallings, *Cryptography and Network Security Principles and Practice*. Prentice Hall, 2nd ed., 1999. ISBN 0-13-869017-0.

[8] T. Koshy, *Elementary Number Theory with Applications*. Harcourt/Academic Press, 2002. ISBN 0-12-421171-2.

[9] IBM Cryptographic Products General Information Manual, Available from http://www-3.ibm.com/security/cryptocards/index.shtml [Accessed 20 April 2002].

[10] P. Kocher, J. Jaffe, B. Jun, and M. Wiener, "Differential power analysis," in *Advances in Cryptology (CRYPTO '99) - 19th Annual International Cryptology Conference*, (Berlin-Germany), pp. 388–397, Springer-Verlag, Lecture Notes in Computer Science, August 1999.

[11] J. J. Quisquater and D. Samyde, "A new tool for non-intrusive analysis of smart cards based on electro-magnetic emissions: the SEMA and DEMA methods," in *Eurocrypt - Rump Session*, 2000.

[12] "Government telecomunications glossary." Available from: http://www.its.bldrdoc.gov/projects/t1glossary2000/ [Accessed 24 April 2003].

[13] P. Smulders, "The threat of information theft by reception of electromagnetic radiation from RS-232 cables," *Elsevier Science Publishers Ltd*, 1990. 0167-4048/90.

[14] NSA/Government Document, *National Security Agency Specification for Shielded Enclosures, No. 24, October 1994*, 94-106 ed., October 1994.

[15] R. Anderson, *Security Engineering - A guide to building dependable distributed systems*. Wiley, 2001. ISBN 0-471-38922-6.

[16] R. J. Anderson and M. G. Kuhn, "Tamper resistance - a cautionary note," in *Proceedings of the Second Usenix Workshop on Electronic Commerce*, pp. 1–11, November 1996.

[17] R. J. Anderson and M. G. Kuhn, "Low cost attacks on tamper resistant devices," in *security protocols - Proceedings of the 5th International Workshop*, (Ecole Normal Supérieure), pp. 125–136, Lecture Notes In Computer Science, Springer, April 1997. 1361.

[18] O. Kömmerling and M. G. Kuhn, "Design principles for tamper resistant security processors," in *USENIX Workshop on Smarcard Technology*, (Chicago Illinois), May 1999. Available from http://www.cl.cam.ac.uk/Research/Security/tamper/ [Accessed 10 December 2001].

[19] "NIST reworks crypto rule." Available from http://www.gcn.com/vol20_no9/news/4075-1.html [Accessed 17 May 2002].

[20] List of NVLAP accredited laboratories. Available from http://csrc.nist.gov/cryptval [Accessed 14 May 2003].

[21] "Security requirements for cryptographic modules." Available from http://csrc.ni-st.gov/publications/fips/fips140-2/fips1402.pdf [Accessed 26 June 2002].

[22] E. Bovelander, "Invited talk on smartcard security." Eurocrypt '97, May 11-15, Konstanz, Germany.

[23] Cryptorgraphy Research, Inc. Available from: http://www.cryptography.com/ [Accessed 10 April 2003].

[24] T. S. Messerges, E. A. Dabbish, and R. H. Sloan, "Investigations of power analysis attacks on smart cards," in *Proceedings of the USENIX workshop on Smartcard Technology*, pp. 151–61, May 1999.

[25] T. S. Messerges, "Using second-order power analysis to attack DPA resistant software," in *Proceedings of Cryptographic Hardware and Embedded Systems - CHES*, pp. 238–251, August 2000.

[26] T. S. Messerges, E. A. Dabbish, and R. H. Sloan, "Examining smart-card security under the threat of power analysis attacks," *IEEE Transactions on Computers*, vol. 51, pp. 541–552, May 2002.

[27] T. S. Messerges, E. A. Dabbish, and R. H. Sloan, "Power analysis attacks of modular exponentiation in smartcards," in *Cryptographic Hardware and Embedded Systems* (C. K. Koc and C. Paar, eds.), vol. 1717, (Berlin, Germany), pp. 144–57, Lecture Notes In Computer Science, Springer-Verlag, 1999.

[28] G. Gandolfi, C. Mourtel, and F. Olivier, "Electromagnetic Analysis: Concrete results," in *Cryptographic Hardware And Embedded Systems - CHES* (C. K. Koc, D. Naccache, and C. Paar, eds.), vol. 2162, pp. 251–61, Lecture Notes in Computer Science, Springer-Verlag, May 2001.

[29] H. J. M. Veendrick, "Short-circuit dissipation of static CMOS circuitry and its impact on the design of buffer circuits," *IEEE Journal of Solid-State Circuits*, vol. 19, pp. 468–473, August 1984.

[30] N. H. E. Weste and K. Eshraghian, *Principles of CMOS VLSI design*. Addison Wesley, 2nd ed., 1993. ISBN 0-201-53376-6.

[31] K. Roy and S. C. Prasad, *Low-Power CMOS VLSI circuit design*. Wiley, 2000. ISBN 0-471-11488-x.

[32] D. Agrawal, B. Archambeault, and J. R. Rao, "The EM side-channel(s): Attacks and assessment methodologies," in *CHES*, Lecture Notes in Computer Science, Springer-Verlag, 2002.

[33] J. A. Muir, "Techniques of side channel cryptanalysis," Master's thesis, University of Waterloo, Ontario, Canada, 2001.

[34] R. Mayer-Sommer, "Smartly analyzing the simplicity and the power of simple power analysis on smartcards," in *Cryptographic Hardware and Embedded Systems - CHES 2000* (C. Koc and C. Paar, eds.), vol. 1965, (Berlin, Germany), pp. 78–92, Lecture Notes in Computer Science, Springer-Verlag, 2000.

[35] L. Goubin and J. Patarin, "DES and differential power analysis - the 'duplication' method," in *Cryptographic Hardware and Embedded Systems* (C. K. Koc and C. Paar, eds.), vol. 1717, (Berlin, Germany), pp. 173–186, Lecture Notes In Computer Science, Springer-Verlag, 1999.

[36] "Introduction to differential power analysis and related attacks." Available from http://www.cryptography.com [Accessed 23 January 2003].

[37] P. Fahn and P. K. Pearson, "IPA: A new class of power attacks," in *Cryptographic Hardware and Embedded Systems* (C. K. Koc and C. Paar, eds.), vol. 1717, (Berlin, Germany), pp. 173–186, Lecture Notes In Computer Science, Springer-Verlag, 1999.

[38] D. E. Knuth, *The art of computer programming - Seminumerical Algorithms*, vol. 2. Reading, Massachusetts: Addison-Wesley, 3rd ed., 1997. ISBN 0201896842.

[39] P. L. Montgomery, "Modular multiplication without trial division," *Mathematics of Computation*, vol. 44, pp. 519–521, 1985.

[40] J. Großschädl, "The Chinese Remainder Theorem and its application in a high-speed RSA crypto chip," in *Proceedings 16th Annual Computer Security Applications Conference(ACSAC'00)*, pp. 384–93, IEEE Computer Society, December 2000.

[41] P. Hofreiter and D. P. Laackmann, "Electromagnetic espionage from smart cards - attacks and countermeasures." Available from http://www.silicon-trust.com/pdf/secure_6/16_tec_update_1.pdf [Accessed 29 July 2003].

[42] F. Bevacqua, E. Cipollone, A. Morviducci, and L. Venditti, "Advances in understanding of em emission from computing devices," in *SECURICOM 7th worldwide Congress on Computer and Communications Security and Protection*, (SEDEP, Paris, France), pp. 9–18, 1989.

[43] R. Lobel and A. Matta, "Collection, treatment, and usage of compromising waves." Available from http://web.mit.edu/dsrobey/www/tempest/tempest.doc [Accessed 29 July 2003].

[44] J. C. Hanisko, "Design engineers battle the dark side of electromagnetism," *Electronic Design*, vol. 48, pp. 99–107, December 2000.

[45] M. Mardiguian, *Controlling Radiated Emissions by Design*. Kluwer Academic Publishers (KAP), 2nd ed., 2001.

[46] H. W. Ott, *Noise Reduction Techniques in Electronic Systems*. Wiley Inter-Science, 2nd ed., 1988.

[47] "TEMPSET timeline." Available from http://cryptome.org/tempest-time.htm [Accessed 26 June 2002].

[48] B. C. Gabrielson, "What is TEMPEST?," in *Senior System Security Engineering Mangagement Seminar*, 1987. Available at: [http://206.102.92.130/ses/papers/TEMPEST/Whatis.html].

[49] Glossary of information warfare terms. Available from http://www.psycom.net/iwar.2.html [Accessed 15 May 2003].

[50] "The complete, unofficial TEMPEST information page." Available from http://www.eskimo.com/~joelm/tempest.html [Accessed 15 May 2003].

[51] "Red/black installation guidance." Available from: http://cryptome.org/tempest-2-95.htm [Accessed 20 June 2003].

[52] John Young's Cryptome site. Available from http://www.cryptome.org/cryptout.htm [Accessed 15 May 2003].

[53] NSA/Government Document: "TEMPEST Countermeasures for Facilities", NS-TISSI No. 7000, 29 November 1993.

[54] "An introduction to TEMPEST." Available from http://www.sans.org/rr/papers/43/981.pdf [Accessed 03 July 2003].

[55] "TEMPEST fundamentals." Available from http://cryptome.sabotage.org/nacsim5000.htm [Accessed 03 July 2003].

[56] TEMPEST letter. Available from http://cryptome.sabotage.org/tempest-cpu.htm [Accessed 03 July 2003].

[57] "The tempest over leaking computers." Available from http://www.ccc.de/mirrors/cryptome.org/tempest-leak.htm [Accessed 10 February 2002].

[58] R. J. Potts, "Emission security," *Information Age*, vol. 11, pp. 151–154, July 1989.

[59] J. A. Lohr, "TEMPEST display systems," *Information Display*, vol. 5, no. 1, pp. 10–15, 1989.

[60] W. van Eck, "Electromagnetic radiation from video display units," *Computers & Security*, vol. 4, no. 4, pp. 269–286, 1985.

[61] P. J. S. Ward, "Computer prediction for radiation security," in *Proceedings of second international conference on secure communication systems*, pp. 6–11, 1986.

[62] "Electromagnetic eavesdropping machines for christmas?." Computers & Security, 1988. Available from http://www.jya.com/bits.pdf [Accessed 03 July 2003].

[63] H. Fang, "Radiation emission from CRT of computer VDU," in *IEEE International Symposium on EMC*, (Washington D.C. USA), pp. 483–487, 1990.

[64] E. Moller, "Protective measures against compromising electromagnetic radiation emitted by video display terminals," *Phrack Magazine*, vol. 4, no. 44, 1990. Available from http://www.shmoo.com/tempest/PHRACK44-11 [Accessed 24 July 2003].

[65] N. E. Koksäldi, S. S. Seker, and B. Sankur, "Information extraction from the radiation of VDUs by pattern recognition methods.," in *EMC'98 Roma International Symposium on Electromagnetic Compatability.*, vol. 2, (Rome - Italy), pp. 678–683, September 1998.

[66] M. Kuhn and R. Anderson, "Soft tempest: Hidden data transmission using electromagnetic emanations," *Information Hiding*, pp. 124–142, 1998. vol. 1525 of Lecture Notes in Computer Science.

[67] R. Anderson and M. Kuhn, "Soft Tempest - an opportunity for NATO," *Information Systems Technology (IST) Symposium*, 1999. Washington DC, USA, 25-27 October 1999.

[68] B. Durak, "Hidden data transmission by controlling electromagnetic emanations of computers." Available from http://abaababa.ouvaton.org/tempest/ [Accessed 10 March 2003].

[69] D. Shiwei, X. Jiadong, Z. Haobin, and W. Changying, "On compromising emanation from computer vdu and its interception," in *3rd International Symposium on Electromagnetic Compatibility*, pp. 692–695, May 2002.

[70] D. Sawdon, "VDU emissions - what, why and standards," in *Electromagnetic Compatibility, Ninth International Conference on (Conf. Publ. No. 396)*, pp. 297–306, September 1994.

[71] H. R. Daneffel, "EMC and COMSEC problems on ISDN subscriber lines," in *ISSLS 88: The International Symposium on Subscriber Loops and Services*, (IEEE, New York, NY, USA), pp. 57–61, 1988. Cat. No. 88CH2536-1.

[72] J. J. Quisquater and D. Samyde, "Electromagnetic Analysis (EMA): Measures and countermeasures for smart cards," in *Smart cards programming and security (e-Smart 2001)*, vol. 2140, pp. 200–210, Lecture Notes in Computer Science, Springer, 2001.

[73] H. Handschuh and P. Paillier, "Reducing the collision probability of alleged comp128," in *Smart Card Research and Application (CARDIS'98)*, pp. 380–385, Lecture Notes in Computer Science, Springer-Verlag, 2000. Vol. 1820.

[74] J. R. Rao and P. Rohatgi, "EMpower side-channel attacks," Tech. Rep. 2001/037, IBM, 2001. Available from http://www.citeseer.nj.nec.com/rao01-empowering.html.

[75] The EM Side-Channel(s): Attacks and Assessment Methodologies (Full report), Available from http://www.research.ibm.com/intsec/emf-paper.ps [Accessed July 25th 2003].

[76] T. Messerges, *Power Analysis Attacks and Countermeasures for Cryptographic Algorithms*. PhD thesis, University of Illinois at Chicago, 2000.

[77] National Instruments Website. Available from http://www.ni.com [Accessed 16 July 2003].

[78] P. Horowitz and W. Hill, *The Art of Electronics*. Cambridge University Press, 2nd ed., 1998. ISBN 0-521-37095-7.

[79] Ultraview Corporation. Available from http://www.ultraviewcorp.com/ [Accessed 10 April 2003].

[80] Gage Applied Inc. Available from: http://www.gage-applied.com/ [Accessed 10 April 2003].

[81] Equivalent Time Sampling. Available from http://wwwasic.kip.uni-heidelberg.de//testlab/tutor/oszi2/oszi_sample.html [Accessed 10 July 2003].

[82] E. English and S. Hamilton, "Network security under siege: The timing attack," *Computer*, vol. 29, pp. 95–97, March 1996.

[83] Microchip Website. Available from http://www.microchip.com [Accessed 10 June 2003].

[84] Farnell Website. Available from http://www.farnell.com [Accessed 23 May 2002].

[85] P. Kocher, "Timing attacks on implementations of diffie-hellman, RSA, DSS and other systems," in *Advances in Cryptology - CRYPTO '96*, pp. 104–113, Springer-Verlag, 1996.

[86] J. F. Dhem, F. Koeune, P. A. Leroux, P. Mestré, J. J. Quisquater, and J. L. Willems, "A practical implementation of the timing attack," in *CARDIS*, pp. 167–182, September 1998.

[87] W. Schindler, "A timing attack against RSA with the Chinese Remainder Theorem," in *Cryptographic Hardware And Embedded Systems - CHES*, pp. 109–124, Lecture Notes in Computer Science, Springer-Verlag, August 2000.

[88] C. Clavier, J.-S. Coron, and N. Dabbous, "Differential power analysis in the presence of hardware countermeasures," in *Cryptographic Hardware and Embedded Systems - CHES 2000* (C. Koc and C. Paar, eds.), vol. 1965, (Berlin, Germany), pp. 252–263, Lecture Notes in Computer Science, Springer-Verlag, 2000.

[89] S. Chari, C. S. Jutla, J. R. Rao, and P. Rohatgi, "Towards sound approaches to counteract power-analysis attacks," in *Advances in Cryptology (CRYPTO '99) - 19th Annual International Cryptology Conference.*, vol. 1666, (Berlin-Germany), pp. 398–412, Springer-Verlag, Lecture Notes in Computer Science, August 1999.

[90] A. Shamir, "Protecting smart cards from passive power analysis with detached power supplies," in *Cryptographic Hardware and Embedded Systems - CHES 2000* (C. Koc and C. Paar, eds.), vol. 1965, (Berlin, Germany), pp. 71–77, Lecture Notes in Computer Science, Springer-Verlag, August 2000.

[91] P. Rakers, L. C. amd Tim Collins, and D. Russell, "Secure contactless smartcard ASIC with DPA protection," *IEEE Journal of Solid-State Circuits*, vol. 36, pp. 559–565, March 2001.

[92] J. S. Coron and L. Goubin, "On boolean and arithmetic masking against differential power analysis," in *Proceedings of Cryptographic Hardware and Embedded Systems - CHES*, pp. 231–237, Lecture Notes in Computer Science, Springer-Verlag, August 2000. 1965.

[93] L. Goubin, "A sound method for switching between boolean and arithmetic masking," in *Proceedings of Cryptographic Hardware and Embedded Systems - CHES*, pp. 3–15, Lecture Notes in Computer Science, Springer-Verlag, August 2001. 2162.

[94] S. Bolger and S. O. Darwish, "Use spread spectrum techniques to reduce EMI," *EDN (US Edition)*, vol. 43, pp. 141–2, 144, 146, 148, May 1998.

[95] Government Document: "Radio Frequency Shielded Enclosures", MIL-HDBK-1195, 30 September, 1988.

[96] F. Bevacqua, E. Cipollone, A. Morviducci, and L. Venditti, "Shielded enclosures design for em data safety," in *International Symposium on Electromagnetic Compatibility*, p. 721, 1989.

[97] J.-J. Quisquater, "Side channel attacks: state of the art.." Available from: http://www.ipa.go.jp/security/enc/CRYPTREC/fy15/doc/1047_Side_Channel_report.pdf [Accessed 23 February 2003].

[98] S. Mercer, "Minimizing rf pcb electromagnetic emission," *R.F. Design*, vol. 22, pp. 46, 48, 55–6, January 1999.

[99] J. Daemen and V. Rijmen, "Resistance against implementation attacks: A comparative study of the aes proposals," in *The Second Advanced Encryption Standard (AES) Candidate Conference.*, March 1999. Available from http://csrc.nist.gov/encryption/aes/round1/Conf2/aes2conf.htm [Accessed 27 March 2003].

[100] D. May, H. L. Muller, and N. P. Smart, "Random register renaming to foil dpa," in *Proceedings of Cryptographic Hardware and Embedded Systems - CHES*, pp. 29–39, Lecture Notes in Computer Science, Springer-Verlag, August 2001. 2162.

[101] E. Oswald and M. Aigner, "Randomized addition-subtraction chains as a countermeasure against power attacks," in *Proceedings of Cryptographic Hardware and Embedded Systems - CHES*, pp. 40–52, Lecture Notes in Computer Science, Springer-Verlag, August 2001. 2162.

[102] C. Clavier and M. Joye, "Universal exponentiation algorithm: A first step towards provable spa-resistance," in *Proceedings of Cryptographic Hardware and Embedded Systems - CHES*, pp. 305–314, Lecture Notes in Computer Science, Springer-Verlag, August 2001. 2162.

[103] M. Akkar and C. Giraud, "An implementation of des and aes, secure against some attack," in *Proceedings of Cryptographic Hardware and Embedded Systems - CHES*, pp. 315–325, Lecture Notes in Computer Science, Springer-Verlag, August 2001. 2162.

[104] M. L. Akkar, R. Bevan, P. Dischamp, D. Moyart, and T. Okamoto, "Power analysis, what is now possible," in *Advances in Cryptology - ASIACRYPT 2000 - 6th International Conference on the Theory and Application of Cryptology and Information Security*, pp. 489–502, Lecture Notes in Computer Science, Springer-Verlag, 2000. Vol. 1976.

[105] N. Rogers, "Killing the EMI demon," *Circuit Cellular*, September 2002.

[106] S. Sebastiani, "Characterization to a tempest testing laboratory and methodology for control to compromising emanations," in *IEEE International Symposium on Electromagnetic Compatibility*, vol. 1, pp. 165–170, August 1998.

[107] J. S. Coron, P. Kocher, and D. Naccache, "Statistics and secret leakage," in *Financial Cryptography*, pp. 157–173, Springer-Verlag, February 2000.

[108] J. Loughry and D. A. Umphress, "Information leakage from optical emanations." Available from: http://applied-math.org/optical_tempest.pdf [Accessed 20 June 2003].

[109] M. G. Kuhn, "Optical time-domain eavesdropping risks of CRT displays," in *Proceedings of the 2002 IEEE Symposium on Security and Privacy*, pp. 3–18, 2002.

[110] National Institute of Standards and Technology (NIST), *Data Encryption Standard*, October 1999. FIPS PUB 46-3.

[111] C. Caldwell, "Why is the number one not a prime?." Available from: http://www.utm.edu/research/primes/notes/faq/one.html [Accessed 12 February 2003].

[112] A. J. Menezes, P. C. Van Oorshcot, and S. A. Vanstone, *Handbook of Applied Cryptography*. CRC Press, 1996. ISBN 0-8493-85237.

APPENDIX A

The Data Encryption Standard

This appendix discusses the DES algorithm. A number of permutations, substitutions and lookup boxes used in the description require tables to represent their operation, however they will not be given here. Any modern book on cryptography will describe them in detail (see [2, 7] for example). The standard describing the DES algorithm is FIPS PUB 46-3 [110]. DES (and most of the other major symmetric ciphers) is based on a cipher known as the *Feistel block cipher.*

As with most encryption schemes, DES expects two inputs - the plaintext to be encrypted and the secret key. The manner in which the plaintext is accepted, and the key arrangement used for encryption and decryption, both determine the type of cipher it is. DES is therefore a symmetric, 64 bit *block cipher*[3] as it uses the same key for both encryption and decryption and only operates on 64 bit blocks of data at a time[4] (be they plaintext or ciphertext). The key size used is 56 bits, however a 64 bit (or eight-byte) key is actually input. The least significant bit of each byte is either used for parity (odd for DES) or set arbitrarily and does not increase the security in any way. All blocks are numbered from left to right which makes the left most bit the LSB.

[3]This contrasts with a stream cipher in which each data element (such as a bit or byte) is encrypted one at a time.

[4]This is a typical block size used in computer algorithms today as it makes attacks difficult to implement but is small enough for efficient manipulation.

Once a plain-text message is received to be encrypted, it is arranged into 64 bit blocks required for input. If the number of bits in the message is not evenly divisible by 64, then the last block will be padded. Multiple permutations and substitutions are incorporated throughout in order to increase the difficulty of performing a cryptanalysis on the cipher [7]. However, it has been stated in [2] that the initial and final permutations offer little or no contribution to the security of DES and in fact some software implementations omit them (although strictly speaking these are not DES as they do not adhere to the standard).

Figure 6.1 shows the sequence of events that occur during an encryption operation (adapted from [7]). DES performs an initial permutation on the entire 64 bit block of data. It is then split into 2, 32 bit sub-blocks, L_i and R_i which are then passed into what is known as a *round* (see figure 6.2), of which there are 16. Each of the rounds are identical and the effects of increasing their number is twofold - the algorithms security is increased and its temporal efficiency decreased. Clearly these are two conflicting outcomes and a compromise must be made. For DES the number chosen was 16, probably to guarantee the elimination of any correlation between the ciphertext and either the plaintext or key[5]. At the end of the 16th round, the 32 bit L_i and R_i output quantities are swapped to create what is known as the *pre-output*. This $[R_{16}, L_{16}]$ concatenation is permuted using a function which is the exact inverse of the initial permutation. The output of this final permutation is the 64 bit ciphertext.

As figure 6.1 shows, the inputs to each round consist of the L_i, R_i pair and a 48 bit *subkey* which is a shifted and contracted version of the original 56 bit key. Details of an individual round can be seen in figure 6.2. The main operations on the data are encompassed into what is referred to as the *cipher function* and is labeled $\boldsymbol{f}$. This function accepts two different length inputs of 32 bits and 48 bits and outputs a single 32

[5]No reason was given in the design specification as to why 16 rounds were chosen.

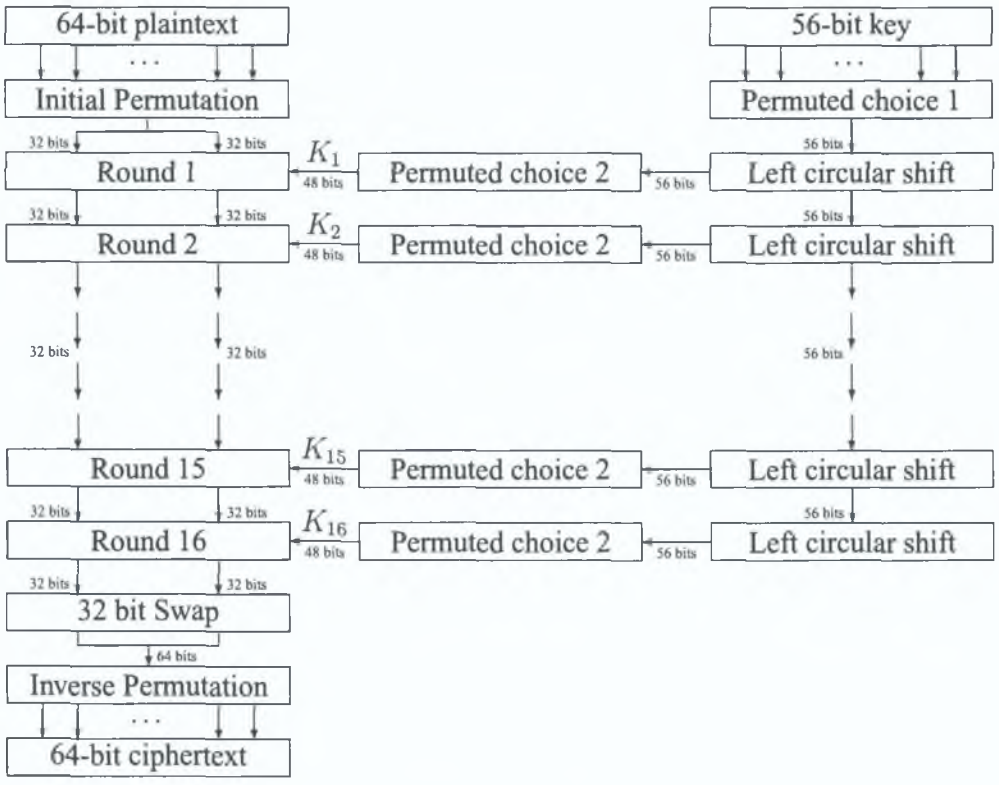


Figure 6.1: Flow Diagram of DES algorithm for encrypting data.

bit number. Both the data and key are operated on in parallel, however the operations are quite different. The 56 bit key is split into two 28 bit halves C_i and D_i (C and D being chosen so as not to be confused with L and R). The value of the key used in any round is simply a left cyclic shift and a permuted contraction of that used in the previous round. Mathematically, this can be written as

$$C_i = Lcs_i(C_{i-1}), D_i = Lcs_i(D_{i-1}) \tag{6.1}$$

$$K_i = PC(C_i, D_i) \tag{6.2}$$

where Lcs_i is the left cyclic shift for round i, C_i and D_i are the outputs after the shifts, $PC(.)$ is a function which permutes and compresses a 56 bit number into a 48 bit number and K_i is the actual key used in round i. The number of shifts is either one

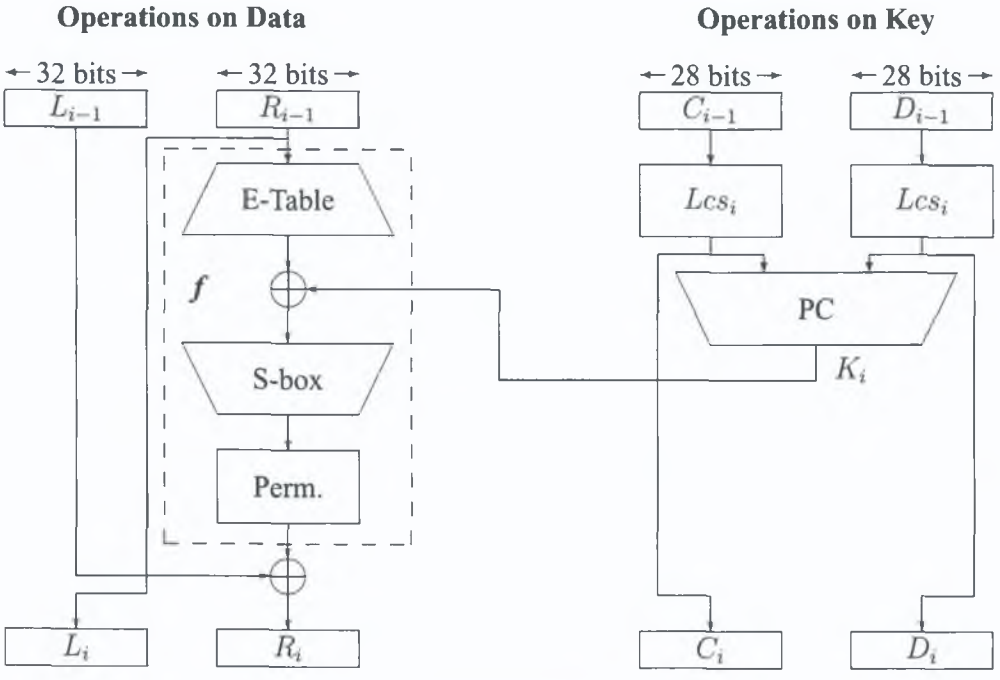


Figure 6.2: Details of a single DES round.

or two and is determined by the round number i. For $i = \{1, 2, 9, 16\}$ the number of shifts is one and for every other round it is two.

The common formulas used to describe the relationships between the input to one round and its output (or the input to the next round) are:

$$L_i = R_{i-1} \tag{6.3}$$

$$R_i = L_{i-1} \oplus \boldsymbol{f}(R_{i-1}, K_i) \tag{6.4}$$

where L and R have their usual meaning and $\boldsymbol{f}(.)$ is the cipher function. This function $\boldsymbol{f}$ is the main part of every round and consists of four separate stages:

1. The E-box expansion permutation - here the 32-bit input data from R_{i-1} is expanded and permuted to give the 48 bits necessary for combination with the 48

bit key. The E-box expansion permutation delivers a larger output by splitting its input into 8, 4-bit blocks and copying every first and fourth bit in each block into the output in a defined manner. The security offered by this operation comes from one bit affecting two substitutions in the S-boxes. This causes the dependency of the output bits on the input bits to spread faster, and is known as the avalanche affect [2].

2. The bit by bit addition modulo 2 (or exclusive OR) of the E-box output and 48 bit subkey K_i,

3. The S-box substitution - this is a highly important substitution which accepts a 48-bit input and outputs a 32-bit number. Figure 6.3 shows the S-box substitution. The S-boxes are the only non-linear operation in DES and are therefore the most important part of its security. They were very carefully designed although the conditions they were designed under has been under intense scrutiny since DES was released. The reason was because IBM had already designed a set of S-boxes which were completely changed by the NSA with no explanation why[6].

   The input to the S-boxes is 48 bits long arranged into 8, 6 bit blocks $(b_1, b_2, \ldots, b_6)$. There are 8 S-boxes $(S_1, S_2, \ldots, S_8)$ each of which accept a 6 bit blocks of data. The output of each S-box is a four bit number. Each of the S-boxes can be thought of as a 4×16 matrix. Each cell of the matrix is identified by a coordinate pair (i, j), where $0 \leq i \leq 3$ and $0 \leq j \leq 15$. The value of i is taken as the decimal representation of the first and last bits of the input to each S-box, i.e. $Dec(b_1 b_6) = i$ and the value of j is take from the decimal representation of the inner four bits that remain, i.e $Dec(b_2 b_3 b_4 b_5) = j$. Each cell within the S-box matrices contains a 4-bit number which is output once that particular cell

[6]Later they claimed that there were certain attacks that they knew about , e.g differential cryptanalysis, which would have been revealed to the public if the design criteria had been exposed.

is selected by the input.

4. The P-box permutation - This simply permutes the output of the S-box without changing the size of the data. It is simply a permutation and nothing else. It has a one to one mapping of its input to its output giving a 32 bit output from a 32 bit input.

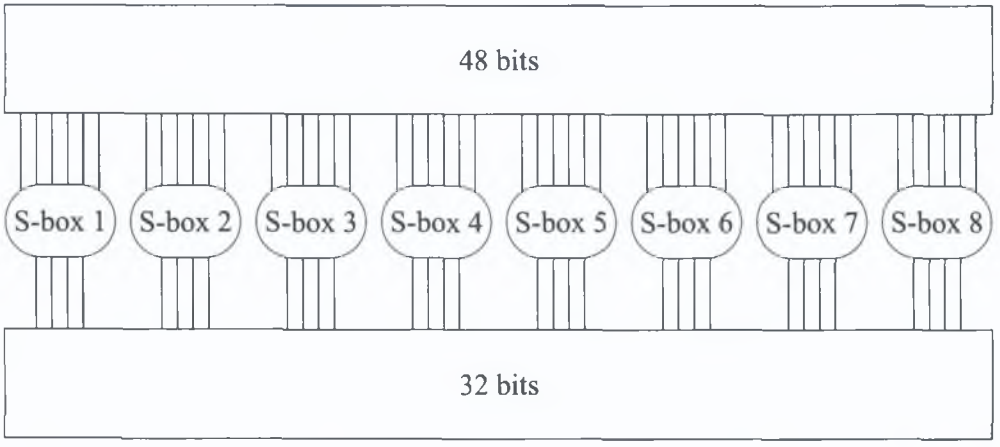


Figure 6.3: Diagram of S-box substitution.

APPENDIX B

Overview of Number Theory

As it's name suggests, number theory deals with the theory of numbers and is probably one of the oldest branches of mathematics. It is divided into several areas including elementary, analytic and algebraic number theory. These are distinguished more by the methods used in each than the type of problems posed. To understand some of the topics discussed in this thesis, a number of elements from these different areas are needed. The relevant ideas are discussed here and include prime numbers, the greatest common divisor, the modulus operator, the modular inverse, Euler's Theorem and Fermat's little Theorem. A rigorous approach is purposely avoided but can be found in [8] if so desired.

Prime Numbers

A prime number p is simply an integer greater than 1 with only two *positive* divisors, 1 and itself. This means that it's entire set of divisors (i.e. its factors) consist only of four integers ± 1 and $\pm p$. It is therefore seen that 1 is *not* a prime number (some reasons for this can be seen at [111]). Prime numbers are of the utmost importance to certain public key algorithms and most of the techniques used will not work without them.

An interesting point to note is that any positive integer $I \geq 2$ is either a prime or can

be expressed as the product of primes[7]:

$$I = P_N^{\epsilon_N} \times P_{N-1}^{\epsilon_{N-1}} \times, \ldots, \times P_1^{\epsilon_1}, \qquad P_N > P_{N-1} > \ldots > P_1 \tag{6.5}$$

or another way of looking at this would be:

$$I = \prod_S P_n^{\epsilon_n}, \qquad \epsilon_n \geq 0 \tag{6.6}$$

where S is the set of all prime numbers[8].

As a result of equations 6.5 or 6.6, any integer > 1 that is not a prime is known as a composite number. It can be seen from this and the definition of a prime number above, that 1 is neither prime nor composite. As an example, the first ten prime numbers are: $2, 3, 5, 7, 11, 13, 17, 19$ and 23.

Division

Any integer can be expressed as $n = q \times m + r$, where n, q, and r are integers, m is a positive integer and $0 \leq r < m$. An important point to note is that the remainder (also known as residue) r, must be *nonnegative* (i.e. either positive or 0). This is seen by two restrictions: $0 \leq r < m$ and $q \times m = \left\lfloor \frac{n}{m} \right\rfloor$[9]. For example, $24 \div 10$ is 2 with a remainder of 4 however, $-24 \div 10$ is -3 with a remainder of 6 and not -2 with a remainder of -4 as might be expected. If $r = 0$ then n is said to be a multiple of m. This is also the same as saying that m divides n, is a divisor of n or is a factor of n and the notation used to express this is $m|n$.

The greatest common divisor, m_{max}, of two integers a and b is the largest *positive* integer that will divide both a and b without a remainder. Therefore, $m_{max}|a$, $m_{max}|b$

[7]This is known as the fundamental theorem of arithmetic.

[8]Clearly in this case however, most of the exponents ϵ_n will be 0.

[9]The notation $\lfloor x \rfloor$ is known as the floor of the integer x and is the greatest integer $\leq x$. Similarly, the notation $\lceil x \rceil$ is the ceiling of the integer x and is the least integer $\geq x$.

and $m_n|m_{max}$ for any divisor m_n of a and b. The notation generally used to represent this is $gcd(a, b) = m_{max}$.

If $gcd(a, b) = 1$, this means that a and b have no common factors other than 1. Such pairs of integers are known as *relatively prime* or *co-prime*. Along with prime numbers, numbers that are relatively prime have considerable importance in public key cryptography as can be seen in appendix C.

The greatest common divisor of two positive integers a and b ($gcd(a, b)$), can be determined by a procedure known as Euclid's Algorithm. It is based on the theorem that $gcd(a, b) = gcd(b, a \bmod b)$ a proof of which can be seen in [7, 8]. It is a fairly simple procedure however as it is not required for and understanding of this thesis, it will not be discussed further here. It is sufficient to know that it exists.

Modular Arithmetic

Modular arithmetic is a form of arithmetic that will generally have been encountered before but may not have been recognised as such. An example given regularly is that of a 12 hour clock, where it is recognised that 3 hours after 11 o'clock it will be 2 o'clock (and not 14 o'clock). Modular arithmetic may seem a little confusing when first encountered but in fact has many parallels with ordinary arithmetic. The symbol used ($\equiv$) is known as the congruence symbol and was invented by the German mathematician Karl Friedrich Gauss around the beginning of the 19th century. It resembles the equality symbol ($=$) quite closely as was likely to be Gauss's intention.

Modular relationships are of the form $n \equiv R \pmod{m}$ (spoken as "n is congruent to R mod m") where n and R are integers and m is a positive integer known as the modulus. If this congruence relationship holds, then it is said that n is congruent to R modulo m. The modulus operator (mod) produces the remainder when the integer on it's left is divided by the modulus. Thus, the term $(R \bmod m)$ is equal to the remainder, r, when R is divided by m.

If two remainders are equal then it can be written that $(n \bmod m) = (R \bmod m)$ - a standard equality. However, if the modulus is equal on both sides of the equation, then the mod m term can be removed from the left hand side and the equality symbol replaced with a congruence symbol (along with a slight rearrangement of the brackets). Assuming $n \neq r$, it would be *incorrect*[10] to say $n = (R \bmod m)$ however it is *correct* to say that $n \equiv R \pmod m$ and this basically states that the same remainder (in this case r) results when both n and R are divided by m.

As was mentioned briefly above, the remainder r is also known as a *residue*. If $R = r$ (i.e. $0 \leq R < m$) then R is known as a *least residue*. The set of numbers congruent to some value $a \pmod m$ is known as a *residue class* (also known as a congruence class). As $0 \leq r < m$, this means there are m possible values of r and hence there are m possible residue classes.

The congruence relationship $n \equiv R \pmod m$ is only true if $m|(n-R)$. To understand why, it must be remembered that the integers n and R can be expressed as $q_{\{n,R\}} \times m + r_{\{n,R\}}$, where the subscript $\{n, R\}$ represents the fact that q and r will generally take on different values for n and R. Only if $r_n = r_R$ will $m|(n - R)$ because in this case the two remainders cancel each other in the $(n - R)$ term: $(q_n \times m + r_n - q_R \times m + r_R) = (q_n \times m - q_R \times m)$. Because $m|(q_n \times m)$ and $m|(q_r \times m) \Rightarrow m|(q_n \times m - q_R \times m)$. If $n - R$ is not divisible by m then the notation used to represent this is $\nmid$ and therefore, $m \nmid (n - R)$. In this case $n \not\equiv R \pmod m$.

Modular Inverse

The idea of an inverse is important both in ordinary arithmetic and modular arithmetic. In any set of numbers, the inverse of a number contained in that set is another number which when combined with the first under a particular operation will give the Identity

[10] Some texts don't use the congruence symbol but instead write $n = R \pmod m$ where the brackets are placed only around the modulus operator and the modulus to identify congruency. This approach will not be used here however.

element[11] for that operation. Two examples of inverses are the additive inverse and a multiplicative inverse.

It must be noted that the Identity element under different operations will be different. For example, under addition it is 0, as any number added to 0 will remain unchanged. However, under multiplication the Identity element is 1 and any number multiplied by 1 will remain unchanged.

In ordinary arithmetic if the number is x then the additive inverse is $-x$ and multiplicative inverse is $\frac{1}{x}$. The idea is the same in modular arithmetic however if x is an integer then its multiplicative inverse would not be $\frac{1}{x}$ as there is no such thing as a fraction in modular arithmetic. In this case it would be a number which, when multiplied by the original number, would give a result that is congruent 1 modulo m (again, m is the modulus).

A number can only have a multiplicative inverse if it is relatively prime to the modulus (i.e., $gcd(x, m) = 1$). This can be seen as one of the reasons why RSA requires its public (e) and private keys (d) to be relatively prime to the modulus as it requires: $e \cdot d \equiv 1 \pmod{m}$

When one number is operated on modulo some other number, it is said that the first number has been reduced modulo the second and the operation is called a modular reduction.

Euler's Theorem

Euler's Theorem which will be stated here without proof (see [8] and [7] for a proof) can be stated mathematically as

$$a^{\phi(m)} \equiv 1 \pmod{m}, \qquad gcd(a, m) = 1 \tag{6.7}$$

[11] This is a number that will leave the original number unchanged under that operation.

where a is any integer and m is the modulus (which, again, is restricted to being a positive integer) The symbol $\phi(m)$ is known as Euler's phi (or totient) function and is the number of positive integers $\leq m$ and relatively prime to it

A few points should be noted about $\phi(m)$

- The value of $\phi(1)$ is defined as being equal to 1
- If p is some prime, then $\phi(p) = p - 1$ as there are $p - 1$ positive integers $< p$ and relatively prime to it
- If p and q are prime numbers and $n = pq$, then $\phi(n) = \phi(p)\phi(q) = (p - 1)(q - 1)$ The reason for this is that the integers *not* relatively prime to p and q are $\{0, p, 2p, \quad , (q - 1)p\}$ and $\{0, q, 2q, \quad , (p - 1)q\}$ respectively The number of integers relatively prime to n is then $pq - (q - p - 1)$, where the 1 is subtracted so as not to include 0 twice

As an example take $4^{12} = 16,777,216 \equiv 1 \pmod 7$ because $7 \times 2,396,745 + 1 = 4^{12}$

Fermat's Little Theorem

Fermat's Little Theorem is really a specific case of Euler's theorem where m is prime[12] It can be stated as follows

$$a^{m-1} \equiv 1 \pmod m, \quad \text{where } m \text{ is a prime and } m \nmid a \tag{6 8}$$

If Euler's theorem is taken to be true, then this can also be seen to work because of the fact that $\phi(m) = m - 1$ for a prime number as mentioned above Again, for a more rigorous proof, see [7,8]

[12]Historically though, Fermat's little theorem was discovered long before Euler s Theorem

APPENDIX C

RSA Algorithm

The RSA algorithm is discussed here All of the required number theory concepts are explained in appendix B The characters used to represent the values are in keeping with most explanations of RSA so although generally m was used throughout this thesis to represent the modulus, n is used here instead

1 Two extremely large, equal length, random prime numbers, p and q, are chosen carefully

2 The number n is found such that $n = p \times q$

3 The Euler phi (totient) function is found from n using the property $\phi(n) = \phi(pq) = (p-1)(q-1)$

4 The value e is chosen at random such that $0 < e < \phi(n)$ and $gcd(e, \phi(n)) = 1$

5 The value d is computed using the congruence relation $de \equiv 1 \pmod{\phi(n)}$ (i e d is the multiplicative inverse of e modulo m)

The ciphertext is then produced by

$$C = M^e \bmod n \tag{6 9}$$

where C is the ciphertext, M is the plaintext and the set {e, n} constitute the public key.

The original plaintext can then be reconstructed by

$$M = C^d \bmod n \tag{6.10}$$

where the set {d, n} constitute the private key and the other variables are as before.

Equation 6.10 uses the fact that

$$de \equiv 1 \pmod{\phi(n)} \tag{6.11}$$

and that

$$C^{\phi(n)} \equiv 1 \pmod{n}, \qquad \text{where} \quad gcd(C, n) = 1 \tag{6.12}$$

which is Euler's theorem from equation 6.7 in appendix B.

This can be seen as

$$C^d \bmod n = (M^e \bmod n)^d \bmod n = M^{ed} \bmod n \tag{6.13}$$

From equation 6.11 it is clear that

$$de = \phi(n)q + 1 \tag{6.14}$$

for some integer q.

Equation 6.13 can then be written as

$$C^d \bmod n = M^{ed} \bmod n = M^{\phi(n)q+1} \bmod n \tag{6.15}$$

which then becomes

$$M^{\phi(n)q} M \bmod n \tag{6.16}$$

which by Euler's theorem can be seen to be equal to

$$M \bmod n \tag{6.17}$$

APPENDIX D

Montgomery's method for modular exponentiation

Practical implementations of theoretical ideas are invariably more difficult to realise than they first appear; this entire thesis is based around this fact. Certain characteristics have to be considered in practice that need not have been considered in theory. Time and space are two major examples as they are both limited resources. The time it takes to run a specific algorithm needs to be optimised as much as possible, as well as the space required to store data values and opcode etc. In 1983, Peter Montgomery proposed a method which optimises the process of modular exponentiation. His paper was published in 1985 [39] and some of the ideas are explained here.

Modular Exponentiation

The process of modular exponentiation is extremely important in public-key cryptography due to it's use in some of the most popular cryptographic algorithms[13]. The general form of such an operation is $b^e \bmod m$, where b is the base, e is the exponent and m is the modulus. The result of this operation is simply the remainder left when $b_1 \times b_2 \times, \ldots, \times b_e$ is divided by m. As discussed in appendix B, this value will always be in the range $0 \leq r < m$.

[13] Including RSA, Diffie-Hellman key agreement and ElGamal.

The obvious method for carrying out $b^e \bmod m$ is to calculate b^e, divide the result by m and then take the remainder. However, it is quite clear that the value of b^e can quickly become very large as e increases - this is the case even for small values of b and m. It must be remembered that if two numbers with s digits in each are multiplied together, the answer will be no more than $2s$ digits long. Similarly, if an s digit number is raised to the power of a t digit number, then the answer will be no more than st digits long. If the values used in RSA are considered (i.e. numbers containing 512, 1024, 2048 etc. bits), then the value of b^e can end up being so big, that it would exceed the value of the estimated number of atoms in the universe! Clearly, no system can store a number of this magnitude, so it would be impossible to work out $b^e \bmod m$ by the method described here.

Luckily, due to the rules of modular arithmetic, it is possible to start with $b \bmod m$ instead of b, multiply this value by itself and reduce the result mod m. This new value can be then multiplied by $b \bmod m$ again, and the result reduced mod m. Continuing in this way for e multiplications, the final value may be obtained without having to deal with numbers larger than twice the number of digits in the modulus (after the initial $b \bmod m$ has been calculated).

For example, consider operating on $129^3 \bmod 7$. As this isn't too large it can be worked out on a calculator to be $2146689 \bmod 7 = 6 \bmod 7$. However, using the technique described, this could be done as $(((129 \bmod 7 \times 129 \bmod 7) \bmod 7) \times 129 \bmod 7) \bmod 7 = (((3 \times 3) \bmod 7) \times 3) \bmod 7 = (2 \times 3) \bmod 7 = 6 \bmod 7$. This would keep the number of digits down to a size that could be handled quite easily. However, as always seems to be the case, in alleviating one problem a new problem is created - there are now far too many divisions.

Division is probably the most computationally expensive mathematical operation and must therefore be avoided as much as possible. In order to obtain the remainder using

the method above, e divisions must be carried out. For most values of m encountered this will be highly inefficient; there is one major exception however.

If m happens to be some power of the radix (or base) of the system being worked in (e.g. 10 in decimal and 2 in binary) then all that is required for a division is a simple right shift of the digits. So, in the decimal system, if the divisor is $125,000$, then to divide by 10^3 say, all that is needed is three right shifts (dropping the zeros of course[14]). Now, computer systems work in binary, and it would not be so easy to divide $125,000$ by 10. However, if m turns out to be some power of 2, then a number of right shifts will suffice (the number being determined by the power). For example, to divide 111111_2 by 2^3 simply involves three shifts to the right (this time the shifted digits are not dropped but are kept as the remainder, i.e. 111_2).

Clearly then, it is desirable to have a way of computing $b^e \bmod m$ by simply right shifting for each division. But how can this be done if the modulus is not a power of the radix? One very popular solution is discussed next.

Montgomery Reduction

Peter L. Montgomery found a way to implement modular division by using powers of the radix of the system, without the need for the modulus to be a power of it. The technique he developed is now known as *Montgomery reduction* and is described in [39, 112] and outlined here for reference.

Basically, given a number X and a modulus m, it is desirable to compute $X \bmod m$ without having to divide by m. Montgomery reduction is the process of computing the value of $XR^{-1} \bmod m$ without having to divide by m. The value R^{-1} is the multiplicative inverse, modulo m, of a chosen integer R. Using this fact, the required value $X \bmod m$ can easily be calculated by $XRR^{-1} \bmod m$.

[14] If the number is not an integer multiple of the base system, then simply dropping the shifted digits will produce incorrect results. However, provided some way is devised to store the shifted digits, a right shift will still produce the correct results.

The value $XR^{-1} \bmod m$ can be calculated using the following algorithm:

```
Montgomery_Reduction(X, R, m, m')
{
    U = Xm' mod R;
    X_red = (X + Um)/R;
    if (X_red > m);
    {
        X_red = X_red − m;
    }
    Return X_red;
}
```

Figure 6.4: This algorithm computes the value of $XR^{-1} \bmod m$ without having to divide by m.

Here, m is the modulus (a positive integer), X and R some integers such that $0 \leq X < mR$, $R > m$ and $gcd(m, R) = 1$, $m' = -m^{-1} \bmod R$ (i.e. m' satisfies the equation $RR^{-1} - mm' = 1$ and is therefore congruent to $-1 \bmod R$) and X_{red} is the reduced value of X. The value of m' can be calculated using the *Extended Euclidean Algorithm* [7, 8].

This technique works because it turns out that

$$(X + Um)/R \equiv XR^{-1} \pmod{m} \tag{6.18}$$

if the above rules are adhered to.

To see this, note that

$$X + Um \equiv X \pmod{m} \tag{6.19}$$

because U is an integer. Multiplying both sides of this by the multiplicative inverse of R gives

$$(X + Um)R^{-1} \equiv XR^{-1} \pmod{m}. \tag{6.20}$$

Now,

$$\begin{aligned}(X + Um)/R &\equiv (X + Um)R^{-1} \pmod{m} \\ &\equiv XR^{-1} \pmod{m}\end{aligned} \tag{6.21}$$

iff

$$R|(X + Um), \tag{6.22}$$

i.e. iff $(X + Um)/R$ is an integer[15]. This can be seen to be the case because

$$\begin{aligned}U &= Xm' \bmod R \\ \Rightarrow Um &= Xmm' \bmod R \\ \Rightarrow Um &\equiv -X \pmod{R} = sR + \tilde{r}_X\end{aligned} \tag{6.23}$$

where s is some integer, $\tilde{r}_X = R - r_X$, and r_X is the remainder when X is divided by R.

Now, for some integer t

$$X = tR + r_X \tag{6.24}$$

and it follows that (X + Um)/R is an integer.

Because $(X+Um)/R$ is an integer, the congruency relationship of equation 6.21 holds and it is clear that

$$(X + Um)/R = XR^{-1} \bmod m + nm, \tag{6.25}$$

for some integer n.

This implies that

$$(X + Um)/R < (n + 1)m. \tag{6.26}$$

[15]It was discussed in appendix B that fractions do not exist in modular arithmetic, so, if $R \nmid (X+Um)$ then it can't be congruent to $(X + Um)R^{-1} \bmod m$ (which is always an integer).

However, due to the restriction $X < mR$, and the fact that $U < R$ (it is the remainder when Xm' is divided by R), this then implies that

$$(X + Um)/R < (mR + mR)/R = 2m. \tag{6.27}$$

Therefore, either

$$(X + Um)/R = XR^{-1} \bmod m \tag{6.28}$$

in which case $n = 0$, or

$$(X + Um)/R = XR^{-1} \bmod m + m \tag{6.29}$$

in which case $n = 1$. The latter case can easily be tested for and if true, the value $(X + Um)/R - m$ returned (this is the purpose of the conditional if statement in figure 6.4).

The above operation will be efficient provided an appropriate value of R is chosen. This should be a power of the radix of the number system being used. For computer systems, it is therefore prudent to choose 2^N (for some N) as the value for R. In order for this value to work, the conditions discussed above must be met (i.e. $gcd(R, m) = 1$ and $R > m$). For practical systems (such as RSA) the modulus is odd and therefore $gcd(R, m) = gcd(2, m) = 1$. The condition $R > m$ will be satisfied if N is chosen to be equal to the number of bits in the modulus, m.

In summary then, the Montgomery reduction of some number X can be stated as

$$X \overset{reduction}{\longmapsto} XR^{-1} \bmod m \tag{6.30}$$

Montgomery Multiplication

Now, as discussed above, the whole point of Montgomery's method is to make modular exponentiation as efficient as possible. Using Montgomery reduction, a special type of multiplication known as *Montgomery multiplication* can be defined. This can be used to allow exponentiation over R using the famous square and multiply algorithm (see figure 2.7 in chapter 2).

The Montgomery multiplication of two numbers P and Q is simply:

$$P \overset{Mont}{\times} Q = P \times Q \times R^{-1} \bmod m \tag{6.31}$$

where $\overset{Mont}{\times}$ denotes Montgomery multiplication[16]. It can be seen that this is simply a Montgomery reduction of $P \times Q$. Therefore, figure 6.4 can be modified to allow this multiplication as follows:

Montgomery_Multiplication(P, Q, R, m, m')
{
$X = P \times Q$;
$U = Xm' \bmod R$;
$X_{red} = (X + Um)/R$;
if ($X_{red} > m$);
{
$X_{red} = X_{red} - m$;
}
Return X_{red};
}

Figure 6.5: This algorithm computes the value of $PQR^{-1} \bmod m$ without having to divide by m.

all values are as before except X, which is now replaced with P and Q. It must be remembered that $0 \leq X < mR$ which means that $0 \leq P \times Q < mR$. This will be

[16] Another way of denoting this might be: **Montgomery_Multiplication**(P, Q, m, m', R) - the name of the function in figure 6.5. However, it is easier to use the symbol $\overset{Mont}{\times}$ with the understanding that the numbers m, m' and R are passed in as parameters also.

satisfied if P,$Q < m$ because then $P \times Q < m^2 < mR\,(R > m)$.

Now, consider two numbers A and B which are to be multiplied mod m. Before carrying out Montgomery's multiplication, both operands need to be transformed into their "Montgomery representation" known as a *Montgomery residue*. This basically consists of performing a Montgomery multiply on each operand with the factor $R^2 \bmod m$. Taking A as the operand to be transformed gives

$$A \overset{Mont}{\times} R^2 \bmod m = A \times R^2 \bmod m \times R^{-1} \bmod m = AR \bmod m \tag{6.32}$$

where $AR \bmod m$ is known as the Montgomery residue of A.

As with Montgomery reduction, the transformation of A and B to their Montgomery residues can be represented as follows:

$$A \overset{residue}{\longmapsto} AR \bmod m = P \tag{6.33}$$

and

$$B \overset{residue}{\longmapsto} BR \bmod m = Q \tag{6.34}$$

Once the transformations are performed, the Montgomery multiplication of the two numbers can take place:

$$\begin{aligned} AR \bmod m \overset{Mont}{\times} BR \bmod m &= ABR^2R^{-1} \bmod m \\ &= ABR \bmod m \end{aligned} \tag{6.35}$$

which equals the Montgomery residue of the product AB.

The fact that the result of equation 6.35 is the Montgomery residue of the product AB is extremely important and is the reason why repeated Montgomery multiplications can be used for exponentiation. The result $ABR \bmod m$ can be used in further Montgomery multiplications until the exponentiation is complete (of course, $A = B$ in this case). If the inputs were not transformed into their respective residues, then the output would be in a different form than the inputs, and repeated multiplication wouldn't work.

Once the calculation is complete, the answer will be of the form $YR \bmod m$, and this must be converted back to the original domain. Performing a Montgomery reduction on this will give the desired result

$$YR \bmod m \overset{reduction}{\longmapsto} YRR^{-1} \bmod m = Y \bmod m. \tag{6.36}$$

Of course a Montgomery reduction is simply a Montgomery multiplication with $Q = 1$ (or $P = 1$). This process is very similar to computing the Montgomery residue of the original numbers with the only change being that a 1 is input instead of $R^2 \bmod m$. So,

$$YR \bmod m \overset{Mont}{\times} 1 = YR \times R^{-1} \bmod m = Y \bmod m. \tag{6.37}$$

If only two integers were to be operated upon using Montgomery's method, then it would actually be uneconomical as the operation takes longer than simply multiplying the two integers mod the modulus. However, the advantage becomes apparent for exponentiation, as the intermediate divisions (which are computationally expensive) are not required.

So, to summarise. Montgomery's method is used for an efficient implementation of

modular exponentiation It does this by allowing the value $b^e \bmod m$ to be calculated over some value R which is computationally efficient to divide with The following steps are taken to do this

1 Compute the Montgomery residue of b $b \overset{residue}{\longmapsto} bR \bmod m$,

2 Use this value as an input to the square and multiply algorithm (in which Montgomery multiplications are used),

3 Perform a Montgomery reduction on the result ($YR \bmod m$) of step 2 $YR \bmod m \overset{reduction}{\longmapsto} YRR^{-1} \bmod m = Y \bmod m$

Index