Technical Report

UCAM-CL-TR-630
ISSN 1476-2986

Number 630

UNIVERSITY OF CAMBRIDGE

Computer Laboratory

Semi-invasive attacks – A new approach to hardware security analysis

Sergei P. Skorobogatov

April 2005

15 JJ Thomson Avenue
Cambridge CB3 0FD
United Kingdom
phone +44 1223 763500

http://www.cl.cam.ac.uk/

This technical report is based on a dissertation submitted September 2004 by the author for the degree of Doctor of Philosophy to the University of Cambridge, Darwin College.

Technical reports published by the University of Cambridge Computer Laboratory are freely available via the Internet:

http://www.cl.cam.ac.uk/TechReports/

ISSN 1476-2986

Abstract

Semiconductor chips are used today not only to control systems, but also to protect them against security threats. A continuous battle is waged between manufacturers who invent new security solutions, learning their lessons from previous mistakes, and the hacker community, constantly trying to break implemented protections. Some chip manufacturers do not pay enough attention to the proper design and testing of protection mechanisms. Even where they claim their products are highly secure, they do not guarantee this and do not take any responsibility if a device is compromised. In this situation, it is crucial for the design engineer to have a convenient and reliable method of testing secure chips.

This thesis presents a wide range of attacks on hardware security in microcontrollers and smartcards. This includes already known non-invasive attacks, such as power analysis and glitching, and invasive attacks, such as reverse engineering and microprobing. A new class of attacks – semi-invasive attacks – is introduced. Like invasive attacks, they require depackaging the chip to get access to its surface. But the passivation layer remains intact, as these methods do not require electrical contact to internal lines. Semi-invasive attacks stand between non-invasive and invasive attacks. They represent a greater threat to hardware security, as they are almost as effective as invasive attacks but can be low-cost like non-invasive attacks.

This thesis' contribution includes practical fault-injection attacks to modify SRAM and EEPROM content, or change the state of any individual CMOS transistor on a chip. This leads to almost unlimited capabilities to control chip operation and circumvent protection mechanisms. A second contribution consist of experiments on data remanence, which show that it is feasible to extract information from powered-off SRAM and erased EPROM, EEPROM and Flash memory devices.

A brief introduction to copy protection in microcontrollers is given. Hardware security evaluation techniques using semi-invasive methods are introduced. They should help developers to make a proper selection of components according to the required level of security. Various defence technologies are discussed, from low-cost obscurity methods to new approaches in silicon design.

Acknowledgements

I would like to thank my supervisor Ross Anderson for his help throughout my research and for encouraging my initial experiments with Static RAM data remanence which in the end reveal much wider problem with data remanence in a wide range of semiconductor memory devices. Without his help and promotion my initial discovery of fault injection attacks would not have become so widely known. His help in proofreading my papers was invaluable. I would like to thank Markus Kuhn for his helpful discussions and Richard Clayton for his help in proofreading. I am grateful to the Department of Materials Science and Metallurgy for allowing access to a FIB machine and fume cupboard, and in particular to Dae-Joon Kang and Nadia Stelmashenko. I would like to thank Hyun-Jin Choi for his help in FIB work. I am very grateful to Specialised Electronic Systems for their help in modelling the fault injection attacks and laser scanning technique. I would like to thank my father Peter Skorobogatov for his helpful discussions on semiconductor physics. The purchase of some equipment used was made possible through the TAMPER hardware security laboratory support provided by NDS and Hitachi. I would like to thank Radiolinija for allowing me access to their semiconductor testing equipment necessary for some of my research. My research was supported by a European G3Card project I took part in. I would like to thank Simon Moore who was the local coordinator of this project in the Computer Laboratory.

Disclaimer

I do not accept any responsibility or liability for loss or damage occasioned to any person or property through using material, instructions, methods or ideas contained herein, or acting or refraining from acting as a result of such use. The reader must be aware of the danger involved in some operations and refer to health and safety warnings for each particular product used. In case of any doubt please seek for professional advice.

The potential hazard involves:

- Chemicals used for decapsulation and deprocessing. They contain very strong acids and alkalines and could cause severe burns to eyes and skin. Adequate protective goggles and gloves must be worn.
- Class 3B laser products used for depassivation and class 3R laser products used for laser scanning and fault injection (visible and invisible laser radiation). Avoid eye and skin exposure to direct and reflected radiation. Lasers can cause permanent damage to eyes and severe burns to skin. Appropriate protective goggles must be worn.
- UV light used for erasing on-chip memories. Avoid eye and skin exposure, as these can damage eyes and skin. Appropriate protective goggles must be worn.

Contents

Chapter 1

Introduction

Semiconductor chips are everywhere around us – from computers, cars, TV sets and mobile phones to mp3 players, washing machines, microwave ovens and phone cards.

With constantly growing demand for security, silicon chips started to be used not only for control purposes but for protection as well. The last ten years have seen a big boom in this area. From military and bank applications, the technology moved to everyday life: to prevent the use of unbranded batteries in mobile phones and laptops, to block non-genuine and refilled cartridges for printers, and to restrict the servicing of your appliances to manufacturer service centres.

These days we have a continuous battle between manufacturers who invent new security solutions learning their lessons from previous mistakes, and the hacker community which is constantly trying to break the protection in various devices. Both sides are also constantly improving their knowledge and experience. In this endless war, the front line shifts forward and backward regularly. Deep down, the problem concerns both economics and law. On the one hand, when dishonest people try to steal property, there will be a demand to increase security. On the other, reverse engineering was always part of technological progress, helping to design compatible products and improve existing ones. The dividing line between legal (reverse engineering) and illegal (piracy) is difficult.

Unfortunately very little attention is paid to proper selection of microcontrollers for secure applications. Mainly this happens because information about the true level of security protection is not widely available from manufacturers or distorted. Chip manufacturers do not pay much attention to the proper design and testing of protection mechanisms. Even where they claim their products are highly secure, they do not guarantee this and do not take any responsibility if the device is compromised. In this situation it is crucial for the design engineer to have a convenient and reliable method of testing secure chips.

In this thesis I try to cover a wide range of problems with the hardware security of silicon chips. As such research is an endless process due to continuous technological progress, I had to choose a narrow area of security analysis in microcontrollers and only slightly touched evaluation of smartcards. Although smartcards offer a greater level of security protection, they are based on the same core design with extra security features added to protect against various kinds of attack. Some attacks, usually invasive, can still be applied to smartcards, but compared

to microcontrollers they will require more effort and therefore would be more expensive and difficult to carry out.

There is no such thing as absolute security. A determined hacker can break any protection provided he has enough time and resources. The question is how practical it would be. If it takes ten years to break a device which in three years time is replaced by a successor with even better security, then the defence has won. On the other hand, the vulnerability could be buried within the design blocks itself. What if your secure system was designed from insecure components? In the end, the overall security of your system is determined by the least secure element. Even if you implement a provably secure protocol, your system could be broken if the key can be easily extracted from the hardware by mechanical or optical probing. Therefore, whenever you design a secure system, proper security evaluation of all the components must be performed. Of course it is impossible to avoid all problems; a reasonable goal is to make the process of breaking your design more expensive and time-consuming. With luck, potential attackers will switch to other products rather than spending money and effort on breaking yours.

It is obvious that one of the first steps in any hardware design is choosing the right components. Despite all the electrical, performance and resource parameters which are widely available from all semiconductor manufacturers, information on security protection and implementation is either limited or totally restricted. That forces the designers to evaluate security-sensitive components themselves or hire other companies for that job. I have tried to give some ideas on hardware security testing throughout my thesis.

A new class of attack – the semi-invasive attack – was recently introduced [1]. Using semi-invasive methods for security evaluation could expose more problems in the hardware design with less effort, and in a shorter time, compared to invasive or non-invasive methods [2]. The archetypal examples of such attacks are fault injection and data remanence, which we are in a process of learning. I would expect more achievements in this area within the next few years.

I have tried to make this thesis acceptable to a wide audience – from embedded designers interested in protection of their intellectual property in microcontrollers, to security analysts in large manufacturing companies. I hope everyone will find something interesting and new in this thesis.

1.1 Previous work and knowledge

Hardware security analysis requires broad knowledge in many areas. Firstly, you have to know the subject of your analysis. That implies, in case of microcontrollers, an ability to write small test programs in assembler and C, and to use development and debugging tools, universal programmers and test equipment like signal generators, oscilloscopes and logic analysers. Performing simple attacks involves using special electronic tools to manipulate the signals applied to the chip. That requires building interface boards and writing programs on a PC for controlling them. More powerful and complex attacks require direct access to the chip surface. For that, some minimal knowledge and experience in chemistry is necessary, as well as the use

of a microscope and microprobing station. Once you get down to the chip layout you need some basic knowledge of silicon design. That comes from microelectronics and could be very challenging task, especially for modern chips. There is a lot of literature available on this subject [3][4] and lots of information on the Internet [5][6], but it takes time to assimilate, especially with deep submicron technologies. Also, the many different design approaches and technological processes used by each individual manufacturer make the analysis more difficult. Common engineering work is required for building special tools and adapters necessary for some experiments. And finally, broad knowledge of physics is needed throughout many of the experiments.

I have tried to cover all these aspects in my thesis, giving the basic idea behind each step involved. As many of the tools required for detailed analysis are either too expensive or not available at all, I had to improvise a lot. This work would be different for well equipped laboratories as they could buy everything they wish. On the other hand, one cannot buy something that has not been built yet or is only being designed. In this case the creative researcher is always ahead as he could build such tools for innovative analysis. Unfortunately academic researchers have very limited funding compared to industrial and government funded laboratories. Therefore only a limited number of invasive methods can be taken into consideration. That was the main reason I paid so much attention to semi-invasive methods as they require much less investment in equipment. Meantime, as it will be seen in this thesis, they give very good results and can be very effective. At the same time, low-cost attacks represent the most severe threat to secure systems as they could be repeated by a large number of individuals. If breaking a secure device requires a multimillion dollar investment, then only a very well-funded laboratory would be able to carry out the work. Even if they succeed, they will be unlikely to give away the information and results for free. In other words, there will be not only technological but economic barriers against attacking such highly secure devices. Also such laboratories usually collaborate, or are directly funded by, large semiconductor manufacturers or governments, and do not do any work for untrustworthy customers.

I travelled a very long way to the hardware security subject. At school, electronics was my hobby. In 1987 I learnt my first microcontroller – Intel 8048. As I did not have any access to a computer or even a programmer device, I wrote my first programs in machine code and then programmed the bytes of the code into a 2716 EPROM chip connected as an external memory to the microcontroller. Of course my programs were quite small, the maximum I did was a programmable timer from one to thousand seconds and a simple calculator. But what I learned from doing such projects was invaluable.

In 1989 I got my first computer – a Sinclair ZX Spectrum 48K. A year later I started building computer external hardware modules like EPROM programmers and ROM emulators, and writing different programs in assembler for the Zilog Z80 processor used in it. All those projects were a hobby, but that was probably the first time I started reverse engineering code. There were two reasons for that. One was to learn the assembler language better through understanding disassembled code; another was to modify the programs, for example, to add extra features. Also I used to modify software drivers to get them working with non-standard hardware equipment. After two years, when IBM PC compatible computers became widely

used, I switched to them. I started at the bottom end – with an XT based on an Intel 8088 processor. I continued developing external hardware devices, but for programming I started using C++ language.

My first security-related project was started in 1995 with testing the security protection in the Motorola MC68HC05 and MC68HC11 microcontroller families against low cost attacks. Some of these attacks were repeated later as a part of my research into non-invasive attacks.

A year later I was reverse engineering the program from a Microchip PIC16C57 microcontroller used in a security access system. The aim of this work was to evaluate how secure the system was from the point of view of non-invasive attacks. It was a door access control system with only two contacts, for an iButton key and an LED for indication. I found some problems that made timing attacks possible, so in the end the design was improved to withstand such attacks. From that I learnt about PIC controllers and started using them for all my further hardware projects.

My Master's thesis project at university was connected with hardware security as well. It involved designing and debugging a cash control monitor – the device used in a supermarket till to store the information on all purchases. As a part of this project, I built a prototype of a PC card with a microcontroller on it and a memory for data storage. Different levels of data access control were implemented inside the microcontroller's program.

After my graduation in 1997, I worked part-time for an ophthalmology company designing microcontroller-based special hardware devices for eyesight testing, training and correction. As these devices had commercial value, I was asked to make them as hard to clone as possible. That was when I got really interested in learning hardware security. In 1999, playing with power glitch attacks, I found a vulnerability in PIC16F84 microcontroller that in the end turned out to be a much bigger problem than simply for that particular device. Only four years later, I realised that it was possible because of the data remanence effect.

Before I came to Cambridge in 2000, all my research on the hardware security of microcontrollers was on non-invasive attacks. Only here at the Computer Laboratory I started learning about invasive attacks and in 2001 I accidentally discovered a new class of attacks – fault injection. We defined this new class of attacks as semi-invasive. By this we mean that, although the access to the chip die surface is required, the passivation layer and the internal wires are not destroyed or physically contacted. This thesis represents the result of almost four years of my research into hardware security.

1.2 The subject of hardware security

Each system has different levels of design with security features normally implemented at each level. The highest level belongs to communication protocols and human interfaces. It might also involve all sorts of restrictions and access control to the building or room where the equipment is running. The application software level supports all the external interfaces and communication protocols and may also do encryption and authentication. It runs under the control of operating system which has built-in security for authentication, encryption and

protection of sensitive information. Highly secure systems, for example, those used in bank applications have hardware tamper resistant modules that keep the secret key inside and perform all critical cryptographic operations. The software communicates with the secure module through the application programming interface (API). This interface is implemented in software running inside the module in a secure environment that prevents anyone downloading or modifying it. The module itself is a hardware device normally implemented as a PCI card for a standard PC board. The program inside this module is stored in battery backed-up SRAM and can be destroyed within seconds if any tampering is detected. The printed circuit board (PCB) with security critical components and sensors against low temperature and ionizing radiation is enclosed within a tamper-sensing mesh and potting material. The whole construction has electromagnetic shielding [7]. In terms of hardware security we will be interested in everything that is located inside this secure module box from the security of the silicon chips buried somewhere on the chip die to embedded software running in the protected environment.

Sometimes it is not quite obvious whether a certain part should be classified as software or hardware. If we have a program running on a PC and providing access to confidential information after entering the right password, then it is obviously software security. If the same sort of access is provided by plugging in a special USB dongle, then the dongle itself is a hardware security device. But what if you have a tamper-resistant secure module with software running inside it? Obviously, there is no firm line between software and hardware, because the first cannot run without the second.

Examples of hardware secure devices are microcontrollers, complex programmable logic devices (CPLDs), field programmable gate arrays (FPGAs), smartcards and secure tamper-resistant modules. They are used in a wide range of applications, from industrial control and wireless communication to bank payment cards, pay-TV applications and building access control.

A good example of hardware security progress is pay-TV [8]. Around 1994 providers started using smartcards for conditional access control. At the same time a large community of hackers was trying to reverse engineer and clone these cards to get free access to the contents of cable and satellite channels. The very first cards were quite simple and had lots of vulnerabilities, like sensitivity to UV light, power fluctuation and reduced clock rates. Service providers learned lessons from this and significantly improved not only the smartcards they used, but the encryption protocols too. Yet after several months determined hackers were again able to find a hole in the security. That again forced the providers into a cycle of hardware security improvements.

Attackers very often are ahead of the defenders for some obvious reasons. Firstly, a single vulnerability could let an attacker succeed, while a defender must protect against all known attacks. Secondly, even if the device can withstand all known attacks, no one can guarantee that a new attack will not be discovered. Initially, security-related bugs are always present in software. The question is who finds one first, and either fixes or exploits it. All these problems force the developer of a secure system to look for better ways of designing and exhaustive testing of his system. If the key element of his system is a standard smartcard, or off-the-shelf

secure chip, he should test it properly, to be sure it will not be broken and reverse engineered in a few weeks time. One way might be to hire a specialised security evaluation company, but the better way, especially for large manufacturers, would be to build his own research laboratory. In this case he will not leak any confidential information, and also do it faster.

This thesis gives some idea of what sort of evaluation can be done by a qualified engineer using widely available and not very expensive equipment. Of course, for proper evaluation of smartcards one will need much more sophisticated equipment, but the basic approach could be the same.

With microcontrollers the first question could be: 'Which chip is the best from the security point of view?' The question is not practical as it will involve testing hundreds of different microcontrollers and in the end the 'best' might not be suitable for the required application. The better approach is to compile a list of microcontrollers suitable for the desired application and then test them against various kinds of attacks. Then you have to find a trade off between the price you will pay for the extra security and the actual protection it will give you. In the end you have to think about who will be likely to attack your device. If they are inexperienced, and the maximum they could do is to try reading the microcontroller used inside your device in a self-made programmer, then most microcontrollers available on the market will provide adequate protection. If you consider your design to be valuable and you think about competitors, then the decision will depend on the amount of money and time attackers could spend trying to reverse-engineer your device. Again, if you worry only about low-cost attacks, then a suitable secure microcontroller could be easily found and you will have to spend some time and money to evaluate it against all known low cost attacks to be sure that nothing will go wrong. If your project is very important and valuable and you are going to invest a lot of effort into algorithm and code development, or you want to provide conditional access control and keep your keys secret, then you must be very careful with your selection. Very likely you will have to choose between different smartcard vendors and not from microcontrollers as they are very unlikely to survive a well equipped and determined attacker.

Another important thing that should be pointed out is that the only practical and reliable way of comparing security in different microcontrollers and smartcards is by estimating the cost of a successful attack which will involve actual breaking into the devices. In other words, a person or a company that carries out the security evaluation should be able to break the security in almost any device within their area of interest. Otherwise the evaluation will be incomplete and theoretical only. In practice it is very difficult to estimate how capable that tester is and how close his report will be to the real situation with the security in a particular device. Another problem that might arise is if he deliberately reports a higher level of security for the tested device in order to prevent you from increasing the security, thus making easier for him to access the information stored in your device later. Unfortunately, very often security evaluation is done in theory based on the information provided by the chip manufacturer and not on proper tedious research. That does not give the real picture of the security and actual cost of any possible attack.

Of course no-one could guarantee that a certain chip will not be cloned or reverse engineered, but proper selection could significantly delay this event. With the speed of technological progress, one year is enough to make an existing product less attractive, and usually it will be either replaced or upgraded. Such upgrade could involve replacement of the security sensitive parts with better versions.

However, as things stand, the security of a standard microcontroller could be significantly improved and some ideas are given in Chapter 8 of this thesis.

1.3 Motivation and overview

This thesis ignores some serious problems in hardware security design of silicon chips. For technical reasons, I had to choose to cover mainly the security analysis of microcontrollers and I only touched lightly on the evaluation of smartcards. I did my best with the equipment I had access to, but it fell far short of what large security analysis laboratories have. Therefore some methods are only mentioned without testing them on actual chips.

I tried to draw a picture of the situation with hardware security in silicon chips from both the design and the evaluation sides. Some low cost defence technologies were suggested together with a possible approach to silicon circuit design that could significantly improve the overall hardware security of chips.

In spite of the poor equipment I have access to, I am very proud of the progress I have done here in the Computer Laboratory during my research. The most important achievement is the discovery of fault injection attacks and defining the new area of attacks – semi-invasive. As such attacks do not require direct mechanical access to the chip surface they are cheaper and easier to apply than invasive ones. Using semi-invasive methods for security evaluation could expose the design problems in days, whereas finding them with conventional invasive methods would take weeks and months, and non-invasive methods might not help at all.

I tried to give as many examples as possible of different security protection schemes in modern microcontrollers. Because the equipment I used for my experiments was too far from modern, I have only very briefly touched hardware security in smartcards, secure ICs, CPLDs and FPGAs.

Chapter 2 of this thesis has some background information about the area of my research. It gives a basic overview of silicon chips' evolution from the technological and hardware security points of view, and information about different types of memory used in microcontrollers. This chapter also discusses the reasons why hardware security is important and popular, and who actually needs it. The last part of the chapter gives an introduction to the crucial part of integrated circuit design – failure analysis. On the one hand, silicon design engineers want to find any design problems that occurred. On the other, if a manufacturer has a tool that allows him to look closely at any point of the chip, then someone else could use it to find a vulnerability in the design or to extract sensitive data.

A general introduction to attack technologies is given in Chapter 3. A classification of attackers and attack scenarios is given, as well as a definition and short introduction into each type of attack. A detailed classification of protection levels is given. This makes the comparison of security in different microcontrollers easier.

Chapter 4 gives examples of non-invasive attacks. These attacks do not require large investments in equipment. It is wrongly assumed that these attacks can be developed by inexperienced hackers with minimal equipment and knowledge. Once found, they can be repeated by almost anyone, but the process of finding the attack could be very difficult and time-consuming. Sometimes one will have to fully reverse engineer the silicon chip to find a vulnerability which could be exploited. This chapter also discusses power analysis techniques, which can sometimes help a lot in understanding the behaviour of a chip. For example, one can easily figure out what command the processor executes, or distinguish between different results of arithmetic operations.

Power and clock glitch attacks are widely used to circumvent many types of protection incorporated in embedded software including smartcards. Also power glitches can be used to break hardware protection and some examples are given in Chapter 4.

Another important area I tried to expose and attract attention to is data remanence in on-chip memories and separate memory devices. This effect could leak a lot of sensitive and secure information after the memory contents are deliberately deleted, by either disconnecting the power supply in case of volatile memories, like DRAM and SRAM, or erasing non-volatile memories, like EPROM, EEPROM and Flash. Finally this chapter presents experiments on data remanence in EPROM, EEPROM and Flash non-volatile memories.

Invasive attacks are discussed in Chapter 5. Starting from an introduction into sample preparation techniques, it gives some idea on what well-equipped and experienced attackers could do with the chip. Although such techniques are well known and straightforward, an attacker will need to have at least basic knowledge about silicon design and be familiar with the different technologies used for semiconductor manufacturing. The main obstacle to these attacks is the cost of the necessary equipment, which could be enormous for attacking smartcards and modern secure chips.

Chapter 6 introduces and discusses the area of semi-invasive attacks. It starts from well known ultraviolet light (UV) attacks, which were counted as invasive attacks before, but actually do not require physical contact to the chip surface or any modifications to the die. Imaging techniques can be counted as semi-invasive as well and they are widely used for partial reverse engineering. Using laser-scanning microscopy one can get significantly more information out of the chip – not only the location of a transistor's active areas, but even its logic state. Both optical imaging and laser scanning can be done through the silicon substrate from the rear side of the chip. This becomes more important for modern chips where multiple top-metal layers that cover the surface prevent us seeing any structures on a die.

Fault injection is another shining example of semi-invasive attacks. With this technique, an attacker can change the state of any individual transistor within the chip. That gives him almost unlimited capabilities in understanding the functions of any block within the design and

circumventing many security protections. Some examples of these attacks applied to microcontrollers are given in Chapter 7. Fault injection can be used to break some encryption algorithms as well. The idea was discussed in the late nineties but no practical means were suggested at the time [9][10][11]. Being synchronised with the system clock and program flow, fault injection can be used as a perfect glitch attack. This is because power and clock glitches usually affect a large number of transistors on the chip whereas fault injection can be focused down to any individual transistor.

Chip manufacturers are constantly acting upon the security problems. Chapter 7 shows some examples of progress in defending against different kind of attacks. Common problems in the security protection design of various chips are also discussed. Technological progress and the continuous shrinking of the chip elements makes invasive attacks very difficult and expensive, and also introduces some problems in implementation of certain semi-invasive attacks. Modern chips with wide data buses, pipelining and instruction caches make power analysis more difficult but not impossible. Different schemes of security implementation are discussed.

Chapter 8 shows various methods of protection against different kind of attacks. There are low budget ways based on obscurity, through to expensive protection techniques requiring quite different approaches to chip design, like asynchronous logic and dual-rail logic.

Finally Chapter 9 summarises the work and achievements of my research. It also proposes directions for further research.

The Appendix contains summarised information on a wide range of microcontrollers and some smartcards.