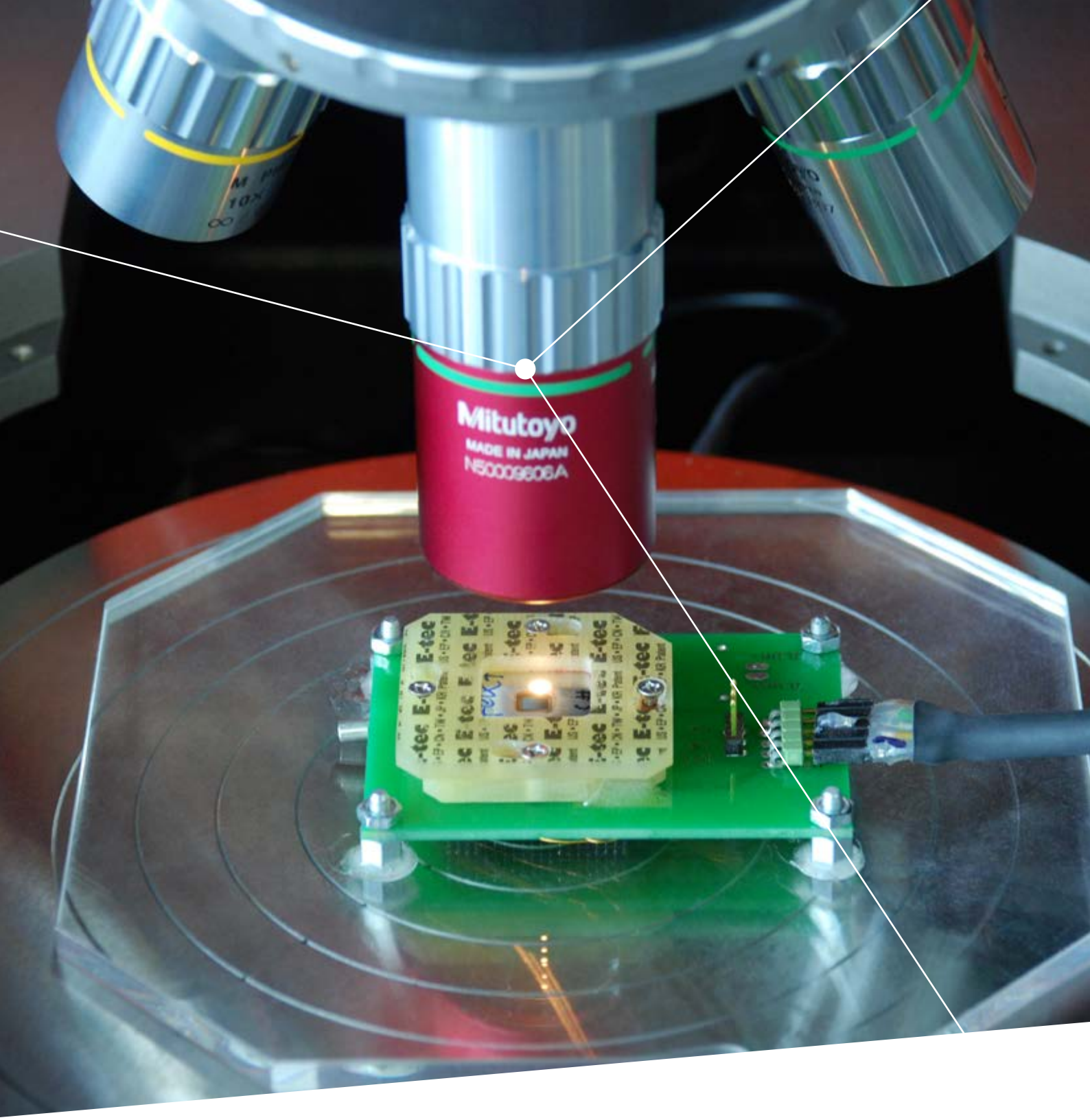
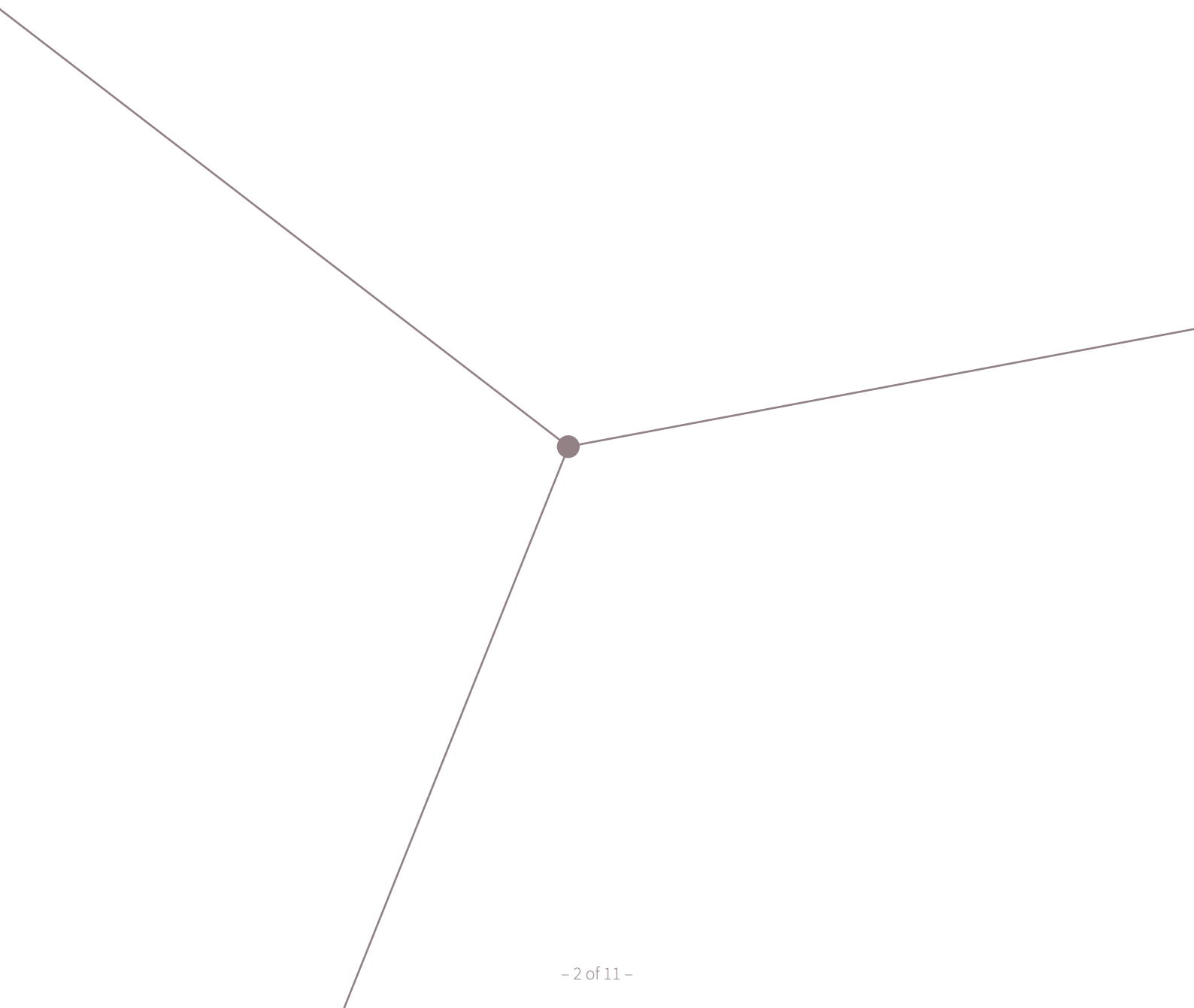


How Secure is Your Chip?

Dr. Peter Laackmann, Marcus Janke

www.infineon.com

Contents



1. Introduction

Back in 1982, when Carnegie Mellon students connected one of the very first smart devices to the internet, they most likely would not have been predicted that today more devices are connected to the internet than people exist on earth. In fact, their computer-connected beverage dispenser can be seen as the predecessor of all the smart factories, home management systems or even driverless cars today. The 'Internet-for-Things', originally coined in the late 90's, quickly evolved to the 'Internet-of-Things (IoT)' nicely showing today's importance of multimodal communication.

IoT permeation means new enterprises, new business models and new revenue streams. But the inherent nature of IoT – connecting 'things' to each other and to 'the cloud' to form a network – gives rise to a variety of new security threats. Therefore, the right security concept should be an integral part of each IoT system design.

Today, all chip manufacturers would have the possibility to access information about modern attack technologies that are used by real adversaries. Even more, a vast selection of hardware security testing equipment is commercially available at a reasonable price. In combination with the appropriate knowhow, these prerequisites allow the development of hardened systems that utilize state-of-the-art countermeasures against attacks.

Nonetheless, still older security approaches are utilized. The reasons are manifold and may include incompatibility of older chip architectures with newer digital security mechanisms, knowhow constraints, or simply the non-existance of security requirements for specific applications. While such approaches may be legit for low-security use cases, on the other hand they may bear significant risks for IoT-connected devices.

It is recommended that an IoT system should at least protect the most critical communication interfaces, plus secrets that are utilized for identification, authentication and encryption. Connected vehicles are a good example in this context. To protect the external communication (e.g. the telematics control unit, central gateway or head unit), robust automotive security controllers are typically used, preferably in combination with an embedded hardware security module. Dedicated security controllers may be used to securely store and process credentials for identification, authentication and encryption purposes (e.g. secret keys).

During the last 30 years of security controller development, a vast variety of security features have been developed and tested – but also many of them have been successfully attacked. Concepts and designs, if they do not originate from a comprehensive security philosophy, may come with a surprisingly short security lifetime. For a customer, chosing the right chip would primarily mean that the underlying security concept, derived from a specific philosophy, should be surveyed before deciding to use such a product in a specific application.

Although today, state-of-the-art countermeasures are readily available, it can be noticed that older security technologies are still utilized. Such practice may include bearing the risk that these products would not meet the requirements of today, thereby risking security failure in the field.

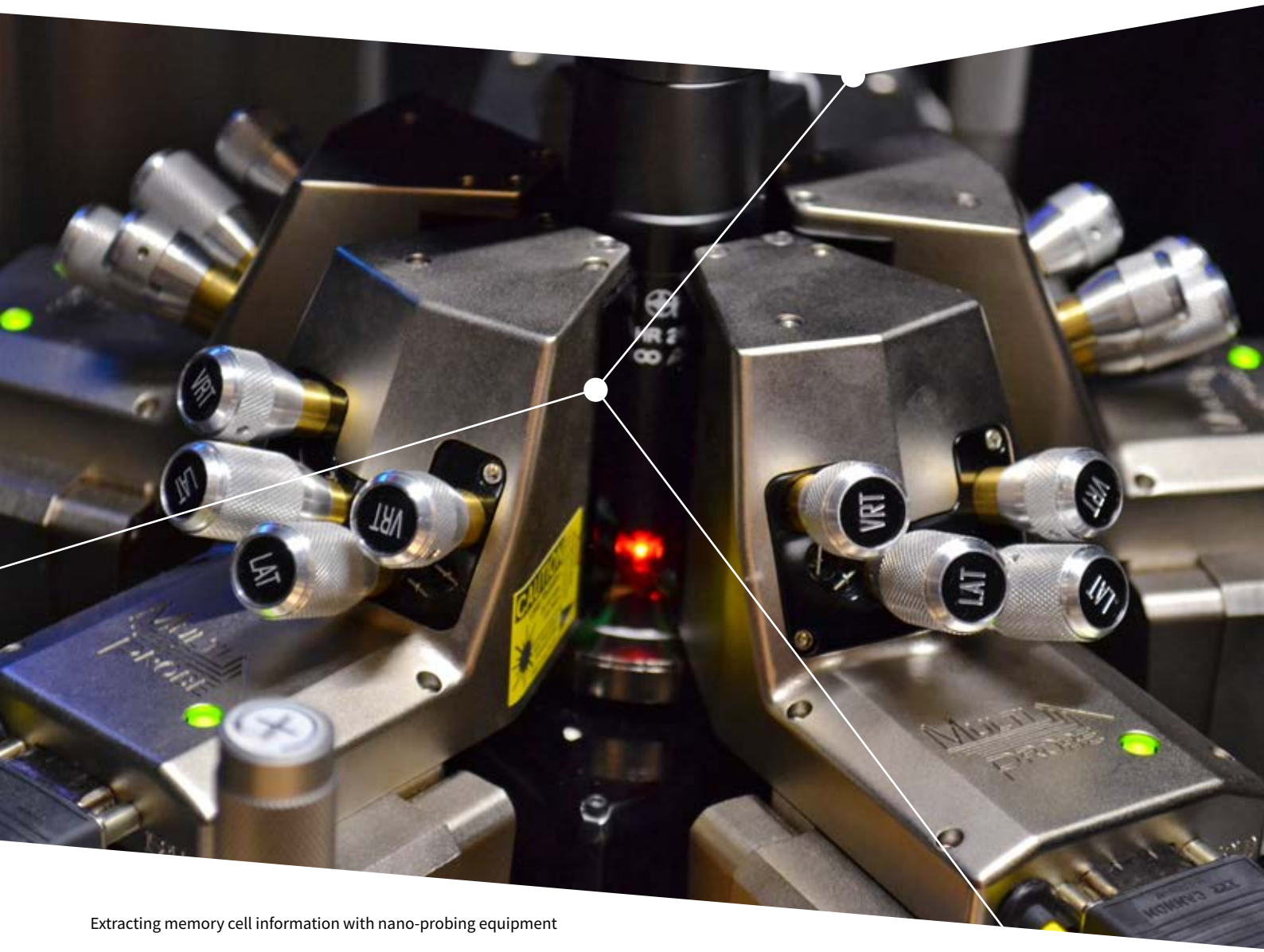
In the following, some of the most important protection measures used for security controllers will be analyzed in detail, explaining the history of protection measures and their remaining efficiency today.

2. Attackers and their goals

In their typical applications, security controllers are utilized for two main reasons: First, secrets like personal keys, data and credentials have to be stored in a way that they are protected from illegal access. In addition, secured storage alone is not enough for effectively protecting secrets. Security controllers, therefore, are also designed to provide a way of securely processing such information.

Attackers, on the other hand, try to extract the valuable information from the chip. Once the information is extracted, an attacker may use that data in order to generate 'emulators', which behave like the original, allowing access, identification or illegal payment. Also, manipulation of chip-internal data is often seen as an important goal for attackers, e.g. in order to change the amount of money stored in a smart card, to change personal data or to forge an identity in an IoT device.

The attackers' efforts and equipment may range from amateur gear to professional devices. Amounts of several hundred thousand of dollars to even millions, spent for attacking security controllers are realistic in the top range. Therefore, security controllers should be up to date in terms of security philosophy, concept strategy and countermeasures for efficiently keeping up with modern attacks.



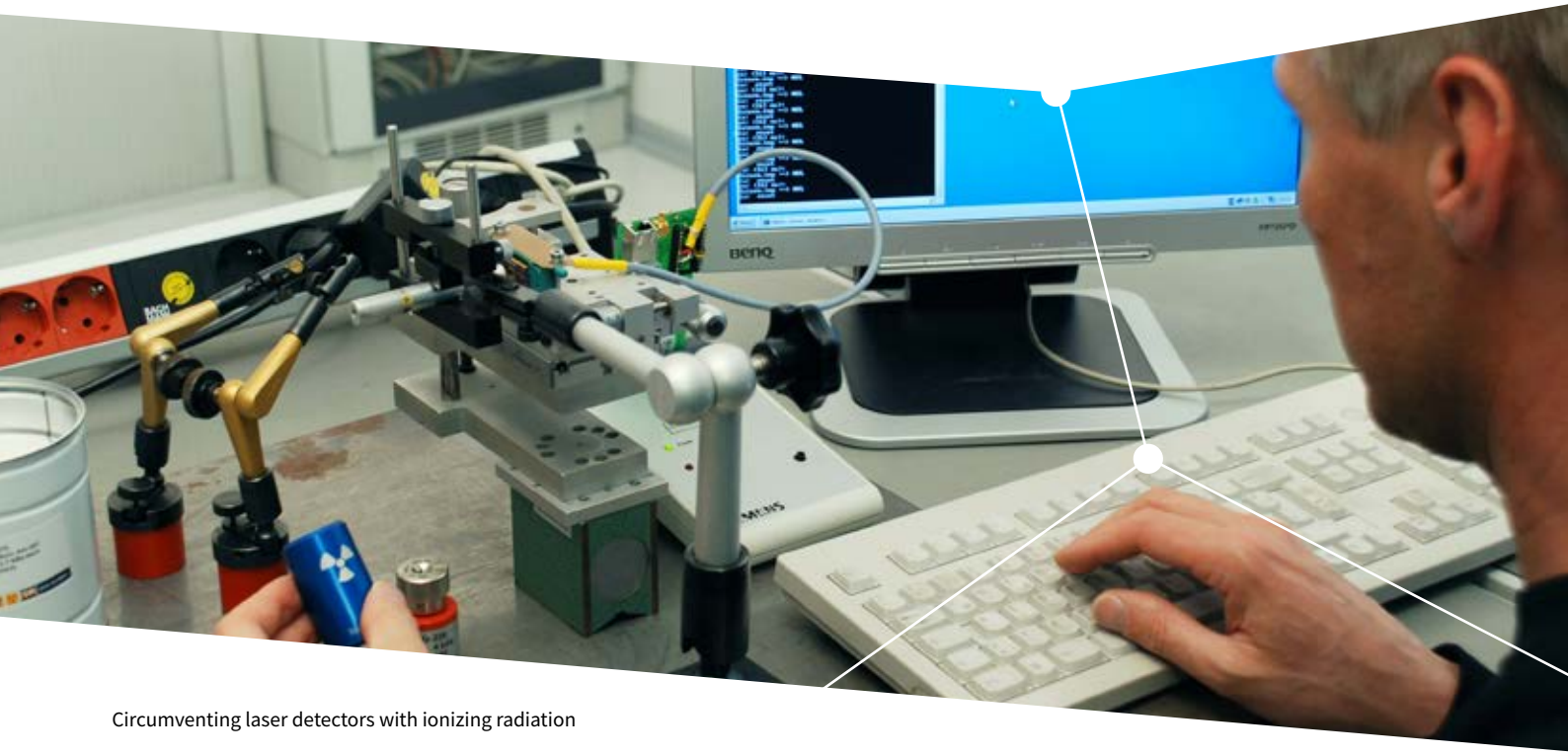
Extracting memory cell information with nano-probing equipment

3. Memory and data path security

Typical security controllers may include FLASH/EEPROM/ROM (non-volatile) and RAM/CACHE (volatile) memory. These memory components and the CPU are connected through the 'bus', transporting adresses and data to and from the memory. More than twenty years ago, it was already seen as daily work for attackers to read out the contents of unencrypted memory from various microcontrollers. The target of these attacks is to get hold on the secret program code and individual data of a security chip. Today, it is common sense that the memory contents of a security controller have to be protected. In the beginnings, manufacturers tried to hide information from attackers by using 'implantation ROM', which uses injection of ions into the chip's substrate instead of visible metal connections for information coding. But, not surprisingly, soon the attackers found out that special etching procedures, called 'decoration', would again allow a good visibility of the memory contents. So the industry, again twenty years ago, moved towards integral security measures instead, and one of the first countermeasures was to scramble the memory adresses. Again, the attackers learned how to break this simple obfuscation, by applying computer programs to reassemble the cleartext from a scrambled memory dump. Even more, the scrambled signals were easily identified and recovered on the chip's layout in those days. Therefore, during the last ten years, simple scrambling of adresses has been extensively replaced by real memory encryption, building an effective barrier against code extraction.

Furthermore, manipulation and fault attacks, which can be used to influence the behaviour of a chip, have to be taken into account. The memory systems, including the correspondent bus structures, are a valuable target for attackers - if not efficiently protected. A successful fault attack could compromise secret keys, dump secured data, circumvent a password or PIN entry or even grant programming access to the chip.

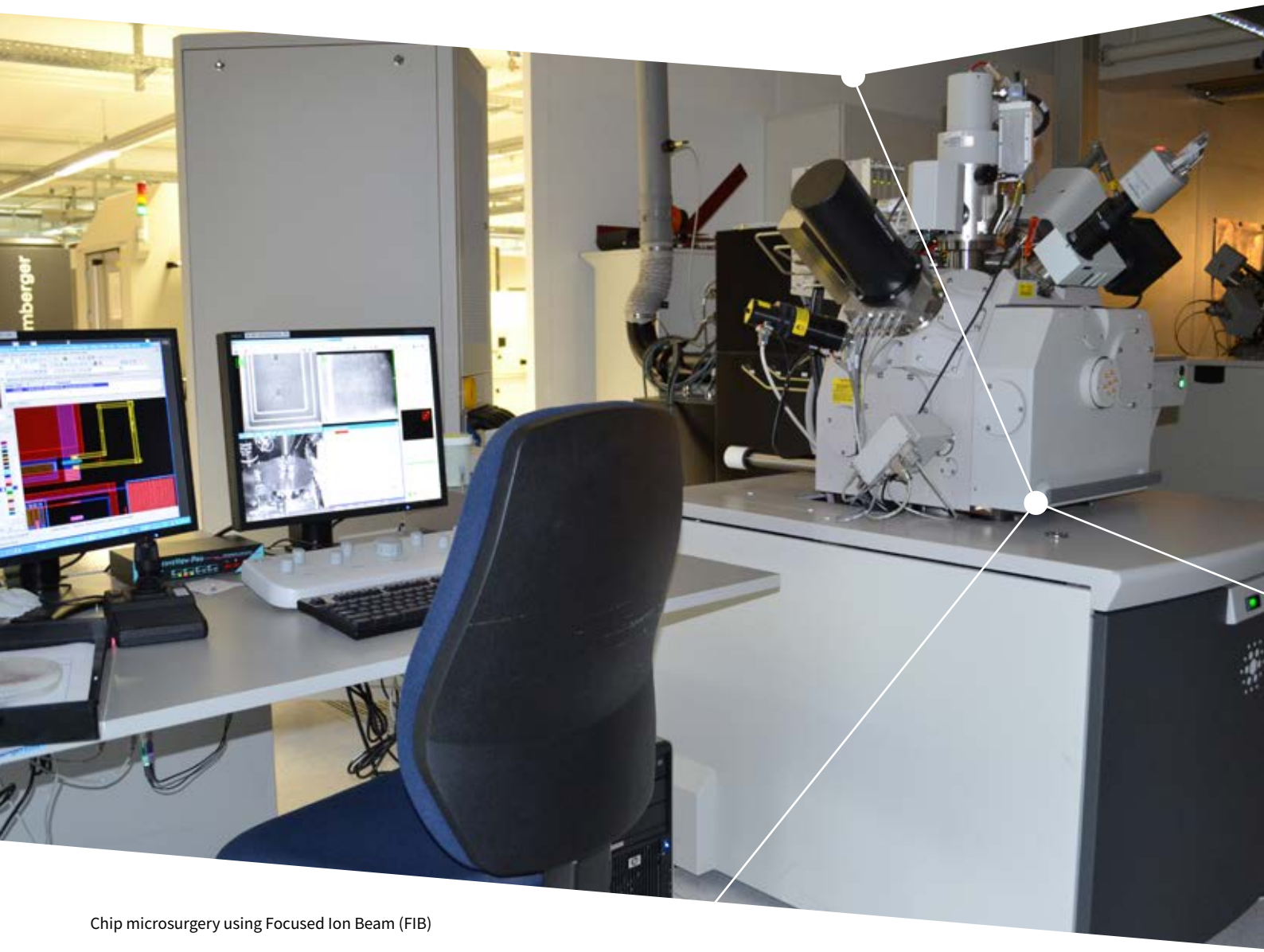
One of the very first protection methods, which may even be used today for low-end security products, is the implementation of so-called 'Parity Bits' or 'Parity Protection'. Using a 'Parity' means that for each memory content part, for example one byte, an additional bit is stored that may be used to check if the sum of bits in a memory word, e.g. are even or odd. It is needless to say that for a vast amount of attacks, the protection level of such parities would be very low – in a typical simple amateur attack experiment, the amount of bits that are manipulated is random, and the probability of a successful 'parity protection' event would therefore just reach about 50 percent. In other words, half of the attacks would then be successful. In fact, few people would accept that as a security barrier. Concurrent systems, trying to detect forbidden states like 'all-zero' or 'all-one', also drastically showed their limitations, as attack methodologies moved to more localized scenarios since 1990.



Circumventing laser detectors with ionizing radiation

Other well-known ways of protecting memory contents from being altered by attackers include mechanisms for secured fetching, that are supported by hardware. Secured fetching is a general term, typically meaning that the memory contents, as delivered to the CPU, are checked for integrity, at least during their delivery – that means before they are processed. As the design implementation possibilities are manifold for such measures, the customer should ask for actual effectivity against attacks as well as for performance penalties, if such features are still used. Of course, it should be avoided that performance has to be traded for security or vice versa. Secured fetching alone may be useless in specific attacks, if only the delivery to the CPU, but not the processing inside the CPU itself would be protected.

Therefore, new methodologies had to be developed and integrated by the chip manufacturers. Due to security reasons, often the use of 'parities' was completely abandoned. Today, mathematical error detection codes are utilized for high-security chips, typically allowing not only error detection of one-bit or more-bit errors, but also error-correction capabilities as by-product.



Chip microsurgery using Focused Ion Beam (FIB)

4. CPU security

Once a memory system is efficiently protected, attackers soon turn to other areas of interest in order to get hold on the valuable data and code stored in a chip, following the data path into the CPU itself. In a typical CPU, the attacker could not only find cleartext during processing, but he could also find excellent opportunities to induce faults during operation, which again could be used for compromising secret keys or manipulate the software execution at will. Early ways of protecting the CPU included sensors outside the core that surveyed the environmental conditions, like voltage, light, or temperature. Unfortunately, most of these sensors usually work to the extent of their localized distribution on the chip. A localized attack on the CPU itself, for example by using lasers, alpha radiation, TIVA (thermally induced voltage alteration) or direct physical forcing attacks like FIB manipulation, 'Nanowhisker-' or AFM needle signal forcing, would therefore still be possible. Today, attacks against the CPU itself are very common, not only in security evaluation or research, but also in the field.

The first solutions to improve CPU security implied the use of parity bits, like already used in the very beginning for protection of memory systems. As parity bits in CPU registers are quite easy to implement, manufacturers do not need to design a security CPU from the scratch – but often, such short-term advantages have to be paid for by strong disadvantages in security performance. Like for the memory systems, one can easily understand that parity protection in the CPU unfortunately comes with very strong limitations in detecting erroneous patterns. Therefore, soon after the design of parity-protected CPUs it was clear that a better way of protecting the heart of the chip itself was desperately needed.

A solution that is often found today is the use of code- and/or data-signatures. These signatures are typically created while compiling the code or generating data, and then stored together with code and data in the memory. During processing, the signatures of the running software are generated in hardware, and meanwhile checked with the reference values stored in the memory. If the values do not fit, an error is indicated and an alarm could be issued, stopping further processing. Code and data signing shows its strengths especially in so-called 'linear' parts of the software code, where no or few branches to other code segments are present. Unfortunately, in many cases, exactly those software parts which use a high number of branches pose the most important targets for attackers, trying to influence the branching behaviour. Nevertheless, code and data signing is still seen as an interesting way to protect mid-range security devices, at least.

A rather simple way of trying to counteract CPU fault attacks was to perform each operation just twice, sequentially. If one of the operations would fail, the hope was that the second could be used to compare the results and state an alarm in case they would not match. Besides hard penalties for the performance, which could be decreased to its half, this method bears the disadvantage that an attacker could try to circumvent the protection by applying a fault attack just twice, too. Today, so-called 'multi-fault attacks' are commonly found in the field, and their relevance in evaluation and certification rises, subsequently.

Since some years, it is recognized that for achieving true CPU protection, one CPU alone would not be sufficient, as it cannot check its own decisions for itself to an extent to efficiently encounter modern attack methodologies. State-of-the-art countermeasures therefore include the use of a closely-interlinked dual CPU core, in which the two CPUs constantly check each other's proper operations and conditions. If the CPUs allow calculation on encrypted data, it is even possible to use different dynamic keys for the two CPUs, further drastically increasing the barrier against so-called multiple fault attacks, using multiple-time and multiple-area fault induction approaches. In contrast, typical sequential solutions, which means utilizing redundancy in the time scale, are expected to fatally succumb to such advanced attacks, especially as appropiate attack equipment components are already available today. Modern fault induction equipment is easily capable of going beyond the time scale of microcontroller instructions and clock cycles, so that the time scale is dissolved and security redundancy in time thus crumbles.

Physical attack targeting an unencrypted CPU

Furthermore, the CPU has to be protected against ‘snooping’ attacks, targeting extraction of information. Interestingly, during nearly three decades of security controller and smart card engineering, CPUs were operating in cleartext. This means that an attacker, who was able to listen on the CPU internals, was also able to reveal the cleartext data processed therein. Early countermeasures included adding several metal layers, that tried to protect the valuable signals from being probed. Later, full-scale chip covering shields were developed, some of which had active self-test capabilities or dynamic random number feeding for checking the shield integrity.

Today, using modern methods like FIB (Focused Ion Beam) chip micro-surgery, attackers showed that such shielding methods have only limited effectivity. Furthermore, there are attack methods to listen to a cleartext-operating CPU that do not need physical manipulation any more, like the long-known, yet re-discovered ‘Optical Emission Analysis’. A silicon transistor emits a very small amount of light while operating, a fact that is used here for this advanced attack path. For example, it could be shown that the contents of registers could be read by using this method.

Again, the definitive need for CPU security improvements could be foreseen years ago. Today’s state-of-the-art countermeasure implies the use of dynamically encrypted data for calculation in the CPU itself, so that the attacker would only yield encrypted data. This method demands CPU design nearly from the scratch, so development is not so easy like for the use of an off-the-shelf CPU design. The efforts, on the other hand, quickly turned out as a major advantage in contrast to conventional approaches.

5. Physical attack protection

The signals running on a silicon chip may be of high interest for an attacker, depending on their relevance. In the eighties, the early days of security controller engineering, attackers already used fine needles to probe and force signals on the chips. During the years, the attack methodologies were strengthened, and today, probing and forcing of signals can be done by micro-surgery on the chip itself, usually by utilizing a FIB (Focused Ion Beam) workstation. In the amateur version of probing/forcing attacks, AFM (atomic force microscopy) probes, or nano-needles made by computer-controlled electrochemical etching of tungsten wires are utilized.

If the (typically) most valuable signals - the CPU contents - are not encrypted, then a high effort has to be spent for protecting the cleartext lines. In the history, this was approached by utilizing synthesized logic, or by hiding security relevant signal lines under a variety of other signals, that are non-critical.

Nearly for two decades, so-called 'shielding' played a major role for protecting critical signals. In the most simple form, just more metal layers are utilized for designing a chip. The target was to hinder an attacker, using amateur equipment, to reach chip signals with critical cleartext data. In the 90's, active shields got common, covering the complete chip surface, thereby hindering visible access and physical attacks. The re-wiring of chip signals, including shielding lines, was taken into account, but rated as highly complex those days.

All historic approaches had advantages and disadvantages, which had to be balanced for each chip family and application. Custom-built design blocks on the chips could be identified more easily than synthesized logic; the latter being an easy subject of automated reverse engineering tools, though.

Today, modern computerized equipment and the possibility of easy access thereto is widely recognized as a threat against many commercial products. Shielding is still applied for products using cleartext-operating CPUs – in this case, still active shielding is used, security optimized wiring or a combination of these methods. State-of-the-art countermeasures use encryption techniques in the CPU itself, so that cleartext is replaced by dynamically encrypted signals.

6. Side channel protection

The useable area of side channel attacks is nearly unlimited – even long before the first security controller has been developed, electrical side channel attacks have been applied to communication devices. Techniques, like originally being used against encrypted teletype devices in the 1960's, have later been transferred to the world of security microcontrollers and smart cards. Today, a vast arsenal of side channel attacks, from timing observation to power analysis (SPA, DPA, PEA, Template Attacks), electromagnetic analysis (EMA, DEMA) and optical emission analysis is available and can be combined with other attack methods for optimal results.

First, side channel attacks were encountered by applying noise to the output signal of a chip, e.g. by use of random power consumption generators. Countermeasures were also achieved by counteracting the chip behaviour, e.g. by employing Dual-Rail Logic, minimizing the generation of side channel information. Today, usually methods are used that combined hardened crypto-coprocessors with certified crypto libraries. One of the latest methods is the use of internally encrypted signals, using dynamic keys, in the crypto-coprocessors themselves.

7. Overview

The following overview table gives an overview on some of the most important protection methods used on security chips, and their recommended use for different application requirements.

	Basic requirements		Highest requirements
Memory (ROM, RAM, EEPROM, Flash) fault attack protection	Parity protection (or NO protection), security sensors	Error Correction Codes (ECC), memory-internal sensors	Mathematical Error Detection Codes (EDC)
Memory and bus snooping protection	Memory address scrambling (or NO protection), Glued/ synthesized logic (or NO protection)	Memory and bus scrambling, Security-optimized wiring	Memory and bus encryption
CPU snooping protection	Glued/synthesized logic (or NO protection)	Glued/synthesized logic and security-optimized wiring	Full data path encryption including CPU encryption
CPU fault attack protection	Parity protection (or NO protection), security sensors	Code/Data Signing, secured fetching systems or flow control checking	Autonomous self-checking (dual) CPU systems
Physical attack protection	Shielding by several metal layers (or NO protection)	Actively controlled shield combined with secured wiring	Encryption, intelligent shielding of control signals
Side channel attack protection (SPA, DPA, EMA, DEMA)	Random noise generators (or NO protection)	Hardened Coprocessors	Coprocessors with internal encryption and CPU encryption
Side channel attack protection (Optical Emission)	Low-Voltage cores (or NO protection)	Low-Voltage cores and small technology node	Coprocessors with internal encryption and CPU encryption

Where to buy

Infineon distribution partners and sales offices:
www.infineon.com/WhereToBuy

Service hotline

Infineon offers its toll-free 0800/4001 service hotline as one central number, available 24/7 in English, Mandarin and German.

- Germany ..... 0800 951 951 951 (German/English)
- China, mainland ..... 4001 200 951 (Mandarin/English)
- India ..... 000 800 4402 951 (English)
- USA ..... 1-866 951 9519 (English/German)
- Other countries ..... 00* 800 951 951 951 (English/German)
- Direct access ..... +49 89 234-0 (interconnection fee, German/English)

* Please note: Some countries may require you to dial a code other than "00" to access this international number.
Please visit www.infineon.com/service for your country!

Mobile product catalog

Mobile app for iOS and Android.

www.infineon.com

Published by
Infineon Technologies AG
81726 Munich, Germany

Date: 09 / 2019

Please note!
This Document is for information purposes only and any information given herein shall in no event be regarded as a warranty, guarantee or description of any functionality, conditions and/or quality of our products or any suitability for a particular purpose. With regard to the technical specifications of our products, we kindly ask you to refer to the relevant product data sheets provided by us. Our customers and their technical departments are required to evaluate the suitability of our products for the intended application.

We reserve the right to change this document and/or the information given herein at any time.

Additional information
For further information on technologies, our products, the application of our products, delivery terms and conditions and/or prices, please contact your nearest Infineon Technologies office (www.infineon.com).

Warnings
Due to technical requirements, our products may contain dangerous substances. For information on the types in question, please contact your nearest Infineon Technologies office.

Except as otherwise explicitly approved by us in a written document signed by authorized representatives of Infineon Technologies, our products may not be used in any life-endangering applications, including but not limited to medical, nuclear, military, life-critical or any other applications where a failure of the product or any consequences of the use thereof can result in personal injury.