Secure Scan Chain and Debug Interface

Von der Fakultät für MINT - Mathematik, Informatik, Physik,
Elektro- und Informationstechnik
der Brandenburgischen Technischen Universität Cottbus-Senftenberg

zur Erlangung des akademischen Grades

Doktor der Ingenieurwissenschaften
(Dr.-Ing.)

genehmigte Dissertation

vorgelegt von

M.Sc.

Frank Vater

geboren am 19. November 1978 in Cottbus

Gutachter: Prof. Dr. rer. nat. Peter Langendörfer

Gutachter: Prof. Dr.-Ing. Heinrich Theodor Vierhaus

Gutachter: Prof. Dr. Michael Gössel

Tag der mündlichen Prüfung: 29. August 2017

DankE.

Contents

Abstract

Cryptographic operations are more and more popular because unencrypted information are a security leakage in many application areas. To read or manipulate encrypted communication by a third party, either the secret key has to be found by brute force attack, but this is mostly impossible for current algorithms, or by extracting the secret key from the device. This is called side channel attack. A possible way to get the secret key is the misuse of test and debug facilities. These interfaces allow a reading and writing access to all internals of the ASIC. Typically they are not protected against a misuse by a third party.

In this thesis a new countermeasure against side channel attacks on scan chain and debug interfaces is proposed. The countermeasure for the scan chain interface, as well as the one for the debug interface, use the same approach, but every interface has its own security component. The approach designed and investigated in this thesis is based on a key matching method, which is resistant against reverse engineering. It is necessary from the user side, to test and debug the device, to write the secret key in an OTP. The "golden" key is embedded in specially designed units, which are used to compare the golden key with one provided in the OTP. After testing or the debugging, the key in the OTP is deleted. Even if this key is known to an attacker, it is not possible to rewrite the value into the OTP. The unit which contain the golden key and compare logic is made of digital standard cells. The cells are not modified, but the wiring has a novelty. Small isolation elements from the analog circuit design are used to implement a "0" or a "1" as value for the golden key. The security feature is the resistance against optical reverse engineering because both types of the golden key and compare unit have the same footprint. Finally 128 of these units compose the 128 bit golden key.

The scan chain solution is suitable for any IP core, independent of whether it is a standalone cryptographic component, a microcontroller or a very complex system on a chip. The additional area is very small and the timing of the circuit is not influenced. For the scan chain test the test pattern generated by the scan pattern generator can be used without any modification. The only requirement is that before the test, the secret key has to be written into the device, and after the test, the secret key has to be deleted.

For a debug interface the same problem exists as for the scan chain interface. An access to the device is an open door for an attacker. As the debug interface is used in different development stages the approach is to implement several OTP lines - one per development stage for example. Additionally a mechanism for different access levels is offered. Depending of the access level different address spaces are unlocked. If required, the implementation can be adapted to specific requirements, e.g., with respect to the number of entries in the OTP or the address spaces.

As shown in this thesis, the solutions for a secure scan and debug interface are easy to integrate into an existing design, while area, timing and power is not influenced significantly. The additional area for a microcontroller is less than $1\,\%$ and the power consumption is increased by an additional leakage of $0.2\,\%$ only. The scan or debug process have to be changed only slightly and the test coverage is not affected and defect analysis is possible. To summarize in this thesis a novel and innovative approach to protect scan and debug interfaces against side channel attacks was designed and evaluated.

Zusammenfassung

Kryptografische Verfahren werden zunehmend eingesetzt, da unverschlüsselte Informationen in vielen Anwendungsbereichen ein Sicherheitsrisiko darstellen. Damit ein Dritter diese verschlüsselten Daten mitlesen oder gar manipulieren kann, muss er im Besitz des privaten Schlüssels sein. Diesen bekommt er entweder per Brute-force-Angriff oder versucht diesen mittels Seitenkanalangriff aus dem Gerät zu extrahieren. Ein möglicher Ansatz den geheimen Schlüssel zu erhalten, ist der Missbrauch von Test- und Debugschnittstellen. Diese Schnittstellen bieten einen lesenden und schreibenden Zugang zu allen internen Speichern eines ASICs und sind in der Regel uneingeschränkt, d.h. auch durch unbefugte Dritte, nutzbar. Somit können auch Schlüssel für kryptografische Verfahren darüber ausgelesen werden.

In dieser Arbeit wird eine neuartige Gegenmaßnahme gegen Seitenkanalangriffe auf Test- und Debugschnittstellen vorgestellt. Der Ansatz basiert auf einem identischen Schlüsselpaar. Um den Chip zu testen und zu debuggen muss des Nutzer den geheimen Schlüssel in einen OTP Speicher schreiben. Dieser wird mit dem goldenen Schlüssel verglichen, welcher im Gerät in speziell entworfenen Einheiten gespeichert ist. Sind beide identisch, wird der Zugriff auf die Schnittstelle gewährt. Nach dem Abschluss von Test oder Debug, wird der Schlüssel im OTP gelöscht. Selbst wenn einem Angreifer nun dieser geheime Schlüssel bekannt sein sollte, ist es nicht möglich diesen nochmals in den OTP zu schreiben. Die Einheiten, welche den goldenen Schlüssel und die Vergleichslogik enthalten, sind mit Hilfe von digitalen Standardzellen konstruiert worden unter Verwendung einer besonders ausgeführten Verdrahtung. Dazu wurden kleine Isolationselemente aus der analogen Schaltungstechnik eingefügt. Mit diesen wird bestimmt, ob die Einheit eine 0 oder 1 enthält. Da beide Einheiten gleich aussehen, ist ein Reverse Engineering mit optischen Mitteln nicht möglich. Insgesamt werden 128 dieser Einheiten in einen Chip eingebaut, so dass der geheime Schlüssel 128 Bit lang ist.

Der Schutzmechanismus kann gleichermaßen für das Scan Chain Interface als auch für die Debugschnittstelle benutzt werden. Jedoch hat jede Schnittstelle eine eigene Schutzeinheit. Die Scan Chain eines jeden Designs und IP Cores kann geschützt werden, unabhängig davon ob es ein kryptografisches Modul ist, ein Microcontroller oder ein komplexes System-on-Chip. Die dafür notwendige zusätzliche Fläche ist klein, die Taktfrequenz und die Leistungsaufnahme werden nicht nachhaltig beeinflusst.

Die Debugschnittstelle eines Microcontrollers wird in verschiedenen Entwicklungsphasen genutzt. Dazu hat der OTP mehrere Einträge, so dass der Schlüssel mehrmals gesetzt werden und die Debugschnittstelle aktiviert und deaktiviert werden kann. Darüber hinaus gibt es einen adressbasierten Mechanimus für die Steuerung von Zugriffsrechten.

In der Arbeit wird gezeigt, dass die entwickelte Lösung für eine sichere Test- und Debugschnittstelle leicht in bestehendes System integriert werden kann. Für einen typischen Microcontroller wird dafür weniger als $1\,\%$ zusätzliche Fläche benötigt. Im Betriebsmodus ist die Leistungsaufnahme nahezu unverändert. Lediglich der Leckstrom wird um 0,2$\%$ erhöht. Außerdem hat der Schutzmechanismus keinen Einfluss auf die Testabdeckung oder die Funktion der Debugschnittstelle. Zusammengefasst wird in dieser Arbeit ein neuer und innovativer Ansatz zum Absichern der Test- und Debugschnittstelle vor Seitenkanalangriffen vorgestellt und evaluiert.

1 Introduction

1.1 Motivation

Dependability, maintainability and security are important subjects of Wireless Sensor Networks (WSNs). Cryptographic algorithms are involved in the data handling today, because it is required to keep data secret and to ensure the data integrity, during wireless transmission.

Many WSNs are in publicly accessible areas, so anyone can steal a sensor node. In case of a stolen sensor node and an extracted secret key for communication purposes, the whole sensor network is at risk. An attacker can get the secret key in different ways; for example Simple Power Analysis (SPA), Differential Power Attack (DPA), social engineering or reverse engineering, or even the use of scan chain and debug interfaces. Per default scan and debug interfaces are not protected and a perfect starting point for attackers. A simple microcontroller is sufficient to start attacks on these types of interfaces. The behavior of scan chain and debug interfaces can be well implemented with a microcontroller. So, it is useful to develop a protection mechanism for it. Especially the requirements of a WSN are the focus of this thesis. Of course, this problem exists in any application area where encryption is used.

Both types of cryptographic algorithms, symmetric and asymmetric, are vulnerable against stolen keys. For example, with the secret key of a symmetric cipher algorithm, all encrypted data can be decrypted. Cryptographic systems are only secure, if the secret key is kept secret. Furthermore, the access to intermediate results of a cryptographic operation should be prevented as well, since such data may be used to reconstruct the secret key. In the best case, the cryptographic component is a black box with well-defined input and output ports and non-interruptible operations. However, this contradicts with the requirements in respect to test, where the access to internal states is mandatory, e.g. via scan interface, to reach a sufficient fault coverage.

The requirements of a secure scan and debug interface are given, because an unprotected circuit puts security at risk for all parties. Interfaces like GPIO or SPI have access to some registers or internal signals only, so that an attack does not have influence to the data, but the scan chain as well as the debug interface controls the ASIC completely. The debug and scan interface have in common, that the attacker needs physical access to the device. Since sensor nodes distributed e.g. onto a field for a farming area, they can get stolen by anyone. Nobody will notice, if one or two missing sensor nodes of a whole network are missing. In a lab, the attacker can prepare the sensor node for his own purpose and re-integrate it back into the network. For this reason, a security mechanism for both interfaces is necessary. In the best case an ASIC has two independent security mechanisms, which are based on the same principal to save effort in development. Due to the fact, that test and debug are two independent tasks, which are performed of independent users, each interface should be accessible separately.

The high number of different debug interfaces in different configurations causes that a design independent approach to secure it, is difficult to develop. In fact, a generic solution such as an additional component is feasible, which controls the access to the debug unit.

In this thesis I developed a new type of protection mechanism for a scan as well as a debug interface for processors, because none of the existing solutions fulfills the requirements. Either they are vulnerable, inflexible, or use to much silicon area or come with severe timing penalties.

Although the solution in this work was developed with focus onto WSNs, it can be used for any other ASIC.

1.2 Imperative of Test and Debug

An ASIC, which is well tested before shipment is necessary because of the complexity of devices today. Several millions of transistors in a simple ASIC are usual, so that a simple functional test after production does not have high test coverage. To reduce manufacturing cost, and to increase the life of the product, a post-production test is indispensable. It took several decades to establish the post-production test as default in ASIC production process. This technical progress should not be stopped, although the first attacks onto ASICs using the scan chain interface are published several years ago.

In the area of microcontrollers almost all devices have a so called "debug interface", but it is used for more than debugging. The second function "in-circuit programming" is required to write a boot loader or an application program to the internal NVM (e.g. flash or FRAM). Without this interface the device cannot be programmed and, so, it is useless. This causes, that a debug interface is an integral part of most processors, which are used in the area of sensor networks. It allows access to a sensor node or even to a completed sensor node network. An unprotected debug interface is easy to use for an attacker and the misuse of the interface is hard to detect. Possible scenarios are the reading out of embedded secret keys or updated selected functions, e.g., simple switch off the encryption mechanism, or even the replacement of the complete operating system.

1.3 Contributions

The objective of this thesis is to introduce a security mechanism for scan and debug interfaces with a high security level. The here proposed solution is resistant against non-destructive attacks and destructive attacks are very hard to perform. The access to the internal data is difficult, even if the device is re-engineered or modified at gate level in a Focused Ion Beamer (FIB). In many known solutions for a secure scan or debug interface a single weak point exists. A single bit (e.g. a fuse) can be repaired or a needle probe influences a signal value to enable the desired functionality.

The approach proposed in this thesis for a secure scan chain interface, is based on the system, which is shown in Figure 1 and it is called "secure scan chain interface" (SSCI). A wrapper contains a security controller, a set of combined golden keys and compare units and memory to store the "secret" key (OTP). The security controller has a programming interface and is used to perform "set" and "delete" of the secret key in the OTP.

To enable access to internal data a matching key pair is required. The golden key has 128 bit and it is stored in the device. Each bit of the golden key is combined with a compare unit, which checks the corresponding bit from the secret key.

The key features of this security mechanism for a scan chain are a secured scan chain interface with a small size and low power consumption. The storage of the golden and the secret key is done on its own secure way. The critical timing of the ASIC is not influenced and the security modules are distributed throughout the layout. The proposed security mechanism of this thesis

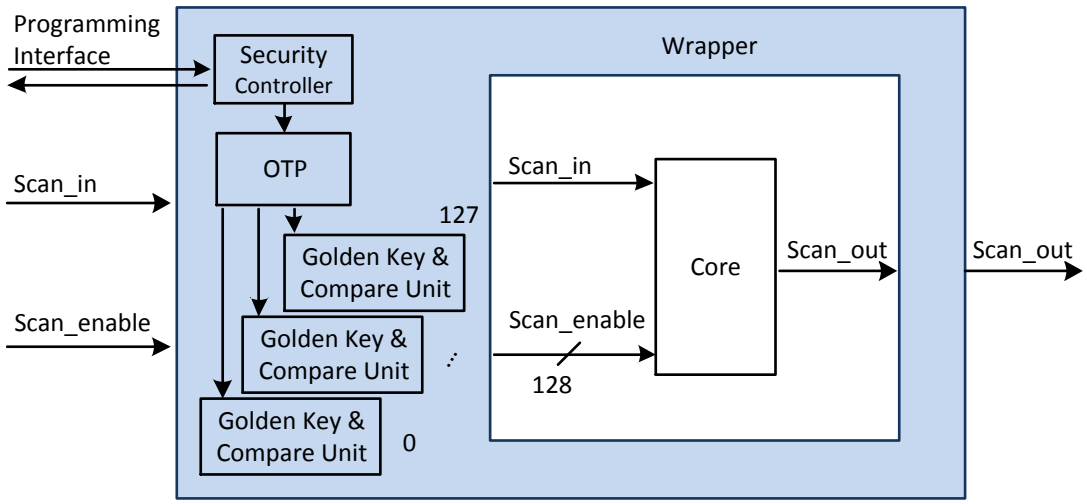


Figure 1: Sketch of the secure scan chain system. A wrapper around the core contains the security mechanism, which is implemented as matching key pair with a key length of 128 bit. Consider the modification of the core. The single bit input of `scan_enable` signal is replaced by 128 bit bus.

is distributed onto the whole die without a single point of vulnerability and it offers a high secure solution. A reverse engineering is hindered because the golden key is stored in a new developed manner. A set of digital standard cells, with a special wiring, combines one bit of the golden key and the compare logic for corresponding the secret key bit, set by the user. Two different sets are necessary: one for the golden key with logical value “0” (CMP0) and one with the logical value “1” (CMP1). Although CMP0 and CMP1 have a different behavior; the layout of both types is almost identical. Only with heavy equipment like a FIB and Transmission Electron Microscopy (TEM) a reverse engineering is possible.

The secure debug interface (SDI) bases on the same approach of a set of golden key and compare units in connection with the OTP. Since a debugging and programming of an ASIC consists of several iterations, the OTP memory was enhanced to set of entries for the secret key. After a finished programming session, the secret key in OTP is deleted and no further programming is possible. As long as free entries in the OTP are available, the secret key can be written into one of them and the program can be updated.

As result of this thesis a new concept of protection mechanism was developed, based on a matching key pair. The compare units which contain the golden key are resistant against reverse engineering. A well prepared implementation allows a fast and safe integration of the protection mechanism in any design. A complex analysis of the impact onto the secured design ensures the expected security level. It was done from transistor level up to system level.

1.4 Publications

For dedicated circuits of this thesis two patents are pending.

- *F. Vater, R. Barth, P. Langendörfer, Chr. Wittke; Patent application 102014226602.5: Halbleiterlogikbauelement mit verschleierter Vergleichslogik; 2014* [84]
  The patent describes a comparator logic, called "compare units" in this thesis built from digital standard cells. Independent, whether the input is tested onto logic "0" or logic "1" the used cells, placement and visual appearance are the same. A MIM structure is used to design the different functionality of the compare units. This MIM element is not visible and so reverse engineering based on optical inspection is impossible. To get a suitable level of security, a set of compare units has to be used.
- *F. Vater, P. Langendörfer, Chr. Wittke; Patent application 102014226606.8: Sicherheitsbauelement zum Zugriffsschutz; 2014* [85]
  The patent describes a security mechanism for access control. Into an OTP memory the secret key is written and the data output of it is connected to the compare units. If the input of all compare units is as expected the access is granted. When the access is no longer required, the key stored in the OTP memory is deleted. Any further access to the protected circuit is not possible, because it is the nature of the OTP that it can be written only once.

1.5 Structure of Work

Chapter 2 introduces the technical background in a wide range. A description of different types of attackers and different types of attacks is given. Furthermore a selection of memory types and camouflaging structures is described. This is the basis for the protection mechanism for scan chain interface as well as debug interface. In Chapter 3 test strategies for ASICs are explained, followed by attacks and possible countermeasures. A similar examination is given for the programming and debug interface. The secure scan chain interface is proposed in Chapter 4 with a detailed explanation and evaluation. A comprehensive security analysis verifies the security mechanism is resistant against different types of attacks. In Chapter 5 the developed countermeasure for the scan chain interface is applied onto the programming and debug interface. This thesis is closed with a conclusion and an outlook for future work.

2 Technical Background

2.1 Attacker Classification

In this section attackers are classified in general. If an ASIC is in the focus of an attacker, it does not matter, in which way the attacker reaches the goal; either reverse engineering by delayering the device and extracting the netlist from taken photographs, or using a scan chain, or debug interface to get some secret information.

Based on an attacker classification the countermeasures can be categorized. For example, an unprotected scan chain can be abused very easily, already by a beginner, to extract information from the device.

1. **Beginner** - For any reasons the beginner is motivated to get information which is not available in a direct way or to modify hardware for a misuse. This class requires a very detailed description how to perform the attack. The equipment is restricted to cheap commercial devices, like microcontroller boards or JTAG programmer.
2. **Independent** - The independent has a restricted budget in most cases, but is highly motivated. Its motivation comes from possible honor, and so the effort of the required work plays an insignificant role. The restricted budget does not imply that he does not have access to expensive measuring equipment. E.g. he could have access to a laboratory at a university or at work for high-end measuring devices.
   The reverse engineering of the MIFARE payment system and the successful breaking of the used cryptographic system [30] is an example for an attack of such class.
3. **Business** - This class is simply driven by finance. In case that the reverse engineering of a product is cheaper than the own development, it is justified that the company uses this way of "product development". Furthermore the reverse engineering is applied on competitive products to search for patent violations.
   In [73] a type of reverse engineering is described, in which an AWACS radar synchronization circuit is replaced. The electronic components on the original PCBs were no longer available, so the whole system was remodeled. The costs for remodeling were $75\,\%$ lower than the estimated costs for manufacturing the original PCB.
4. **Government** - The government becomes a hacker, if the national security is at risk. The security has a high priority so that the financial budget might be limited, but the limit is very high. Mainly the decryption of cipher text might be the main focus of this type of attacker. Time would be a limiting factor because after a certain time the information is not required anymore.

2.2 Test and Tamper-resistant Hardware

The FIPS140-3 document [57] describes the certification process and its security requirements for cryptographic modules and four security levels are introduced. Cryptographic modules are defined as hardware, hardware and firmware, or software. Even the testing of modules is part of the certification process. On the one hand the document postulates a self-test (pre-operational and periodic during operations), on the other hand a vendor test is necessary to exclude production errors.

To reach security level 1 and 2 it is sufficient to perform a functional test of the cryptographic module. In case of an AES implementation it would be sufficient to perform one encryption and one decryption of a sample vector, which is given in AES validation suite [40]. To get a certificate for level 3 and 4 a "low-level testing" is necessary and this low-level testing is restricted to hardware modules only. The document does not give any statement, how the low-level test is implemented. E.g. for the AES, each functional module (AddKey, SBox, ShiftRow, MixColumn) can be tested separately. A set of known input vectors are applied to each module and the response is compared to a golden vector. Such a test has to be implemented and the implementation has to be verified. Another approach of a low-level testing is a scan chain. Such a scan chain insertion is supported by different tools and is independent of the current implementation. A well implemented scan chain reaches a test coverage of more than $99\,\%$, which guarantees a low-level test.

In FIPS140-3 document [57], the details for a secure design are described. Level 3 and 4 are only reachable for hardware based implementations, so these levels are described more in detail here. The physical requirements for level 3 are:

- Tamper response and zeroization circuitry on removable covers
- Protection from probing, e.g. needle probes on open device
- Hard opaque coating

The physical requirements for the highest security level, level 4, are additionally to level 3:

- Tamper detection and zeroization circuitry for multi-chip modules
- Fault injection mitigation

Especially the requirement "fault injection mitigation" for level 4 shows that the aim of a tamper-resistant design is not to make an attack impossible, because no one can guarantee a fully tamper-resistant device. In fact the requirement is to harden a design against attacks.

An attacker of class 1 (beginner) should not be able to get access to device internals. More difficult to assess is an attacker of class 2, but even such an attacker should not be able to get access to internal data of a protected device. For professional attackers (class 3) a certain security level is required. The developer team of the tamper-resistant device should estimate the cost for a successful attack and in any case an attack should cost much more money than a third party is able to earn. Clearly such estimation is very difficult because only known attacks can be considered. In [51] an overview of reverse engineering techniques is given, which is related to consumer electronic systems and not to a tamper-resistant ASICs. But it can be seen as a starting point and the approach of estimation can be derived for ASICs.

Attackers of class 4 are an exception. They are supported by their government, so that in most cases money is no object. Here it is more a question of time instead of money to get the secret data. Furthermore, if the technical approach collapses, other ways such as social engineering seems to be a suitable way to get the required information. Most successful attacks are exploiting "social engineering", and maybe even the cheapest attacks. For example, in [48] a study was published in which pedestrians were offered a piece of chocolate if they gave away their password. Up to $50\,\%$ of the participants wrote down the correct password onto the questionnaire.

2.3 JTAG

JTAG is a well-known interface for testing and debugging. In the IEEE 1149.1 standard [32], the input, the output and the behavior of the serial interface are described. It can be used for programming and debugging of microcontrollers, to perform a scan chain test, or even to initiate a BIST or a memory test - i.e. it is a flexible interface for testing and debugging purposes with a low pin count.

This type of interface is very common; therefore many IP cores are available. E.g. openJTAG [64] offers an open source implementation in a HDL, or the company Synopsys has a configurable JTAG module [76]. The open source solution has the advantage that the user is able to review the code and to adapt in regard to its own requirements. Any unused functionality can be removed with impact on area and power consumption. The DesignWare component from Synopsys is a black box, which can be configured by the user within limits.

2.3.1 Test Capabilities

A useful feature of the JTAG interface is that several ASICs can be connected in one chain and one physical access port (e.g. pin header on PCB) is sufficient for data transfer. In Figure 2, an example of three connected ASICs is given. The devices are called Test Access Port (TAP). One single TAP can be an ASIC on a PCB or even several dies inside of a package.

An internal JTAG controller has a set of different registers: an instruction register, a bypass register and a user data register. Each register is accessible from the JTAG interface as shown in Figure 3 and it is linked to certain commands and functions.

The different types of commands allow different tests, for example “intest” and “extest” mode. Since the “intest” mode is responsible for testing the ASIC internally, the “extest” is responsible to test the board connections. In Figure 3 the required boundary scan registers (BSC) are shown, which are directly connected to the pad. The boundary scan registers are a replacement for needle test points on the PCB. The test points are now available via a communication interface instead of a large amount of needle probes. Also if the internal test is disabled, the test of the board connections can be performed. The board connections can be tested at a low frequency only, because the data is transferred from one boundary scan register in one ASIC to the boundary scan register in the connected ASIC.

The “intest” allows a scan-based test and to execute the test code on the microcontroller for diagnostic purposes. The patterns for the scan chain tests are generated by a pattern generator.

2.3.2 Debugging Capabilities

Two different approaches, shown in Figure 4 and Figure 5, are possible to connect a CPU core to the JTAG interface for debugging. The first one in Figure 4 is used in devices of ATMEL [3]. All flip flops are connected like a scan chain. During a debug session, all data of all registers are read out and displayed in the debug environment. For a large shift register, it takes a long time to read out the content. Furthermore, without any additional effort such a solution is applicable on designs with one scan chain only. The advantage during a debug session is that all registers are read out and displayed in the debug tool.

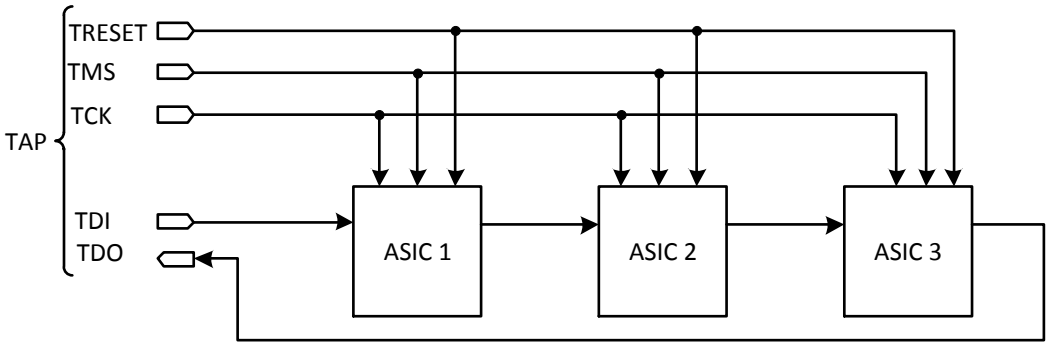


Figure 2: Schematic of several ASICs, which are connected in a JTAG chain. TRESET, TMS and TCK are shared pins; TDI is the input to the first device and feed through to all other devices. The output of the last ASIC in the chain is connected to the TDO pin.

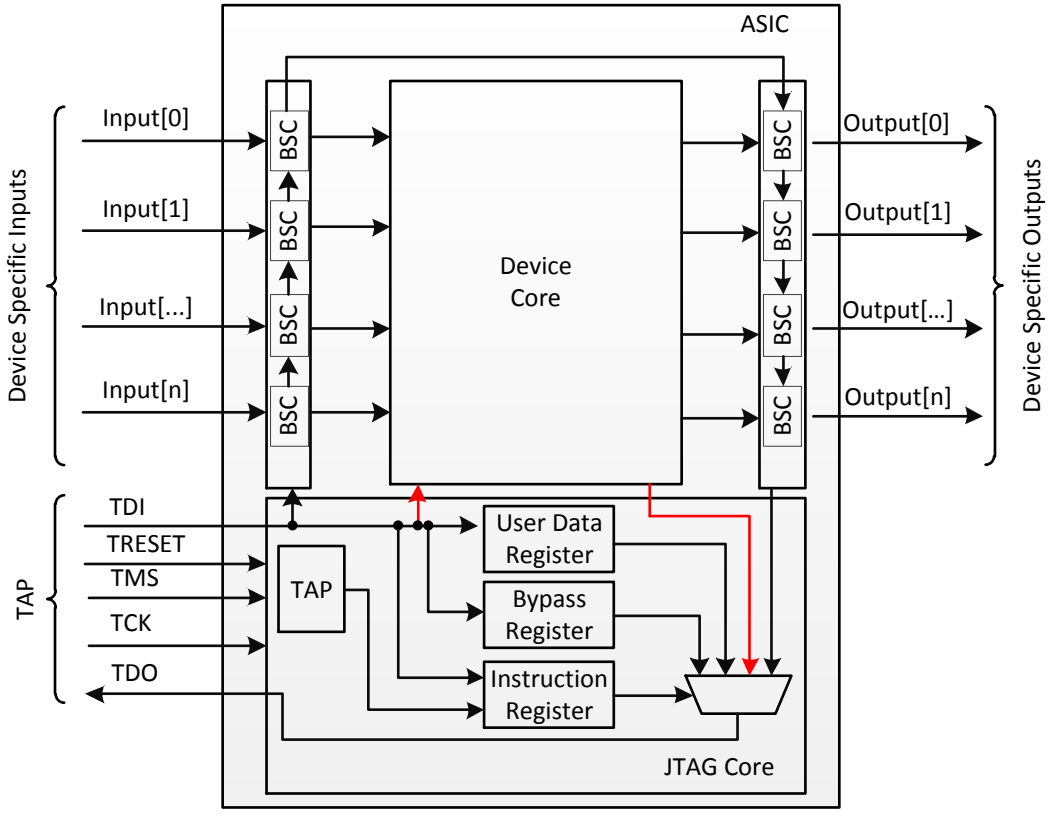


Figure 3: ASIC with additional, simply JTAG component and its interface. JTAG is used for boundary test and scan chain test (red arrows) in this example.

The second approach in Figure 5 is a separate unit for debugging which has access to the data bus of the microcontroller. An example of this solution is the MSP430 from TI [80]. Several different commands control the MAB, MDB, CPU behavior and further settings. As result, the command from the serial interface is translated to a parallel access onto the internal bus structure. For a “read” or “write” access to a specific address or register, a set of commands has to be sending per JTAG interface into an instruction register. A “read” or “write” request to a register within designs with a high number of flip flops is proceeded much faster. To get the data from a specific address, only the data from these flip flops have to be read. In contrast to this solution, in the first solution all flip flops have to be read, which is much more time consuming.

2.3.3 Security Solutions

In both selected examples for the use of a JTAG controller as a debug interface in microcontrollers, the Atmega32 as well as the MSP430, have security features. According to the JTAG standard, a set of user-defined registers is allowed, which offers the possibility to implement functions e.g. to support a password protection, setting of fuse bits or an authentication process. But the implementation of hidden commands should be avoided. In [68] it is shown that they are security risks.

2.4 State of the Art - Attacks

2.4.1 Introduction

An attack on a sensor node or any other system can be driven by different aspects. In rare cases the attacker can be the owner of the device itself, to extract a firmware for which the sources are lost. Even an attacker from outside can be interested in the firmware of the device, e.g. to save development time or to get efficient software algorithms or implementations. The software image can be disassembled, so a human readable program can be generated.

The node itself can be very expensive, so that an attacker might be interested to get full control of the node and to misuse a stolen node for its own application area. Also the extraction of a secret key of a cryptographic algorithm can be in focus. Such a stolen key can be helpful to decrypt stolen measurement data or to manipulate it, which can be risky e.g. in agriculture applications. Wrong measurement data can result in wrong watering or manuring, so that the plants might be damaged. If pictures, taken by a digital camera, get extracted by an attacker, privacy is at risk. Measurement data or the program itself can be affected on-the-fly by a debug interface, which is connected to a host.

It does not matter, what the exact goal of an attacker is. In fact, a sensor node which is under control by a third party, puts the whole network at risk. The effort to secure a device causes additional costs to the owner. It turns out to be cheaper to invest in a secure node and network, instead of running the risk to get one’s network invaded by an attacker.

Knowledge about different types of attacks is necessary to develop suitable countermeasures. For example, a side-channel attack can be used to extract a secret key from a device, but the secret key is used to protect a debug interface. Now it is possible to get full access to the device.

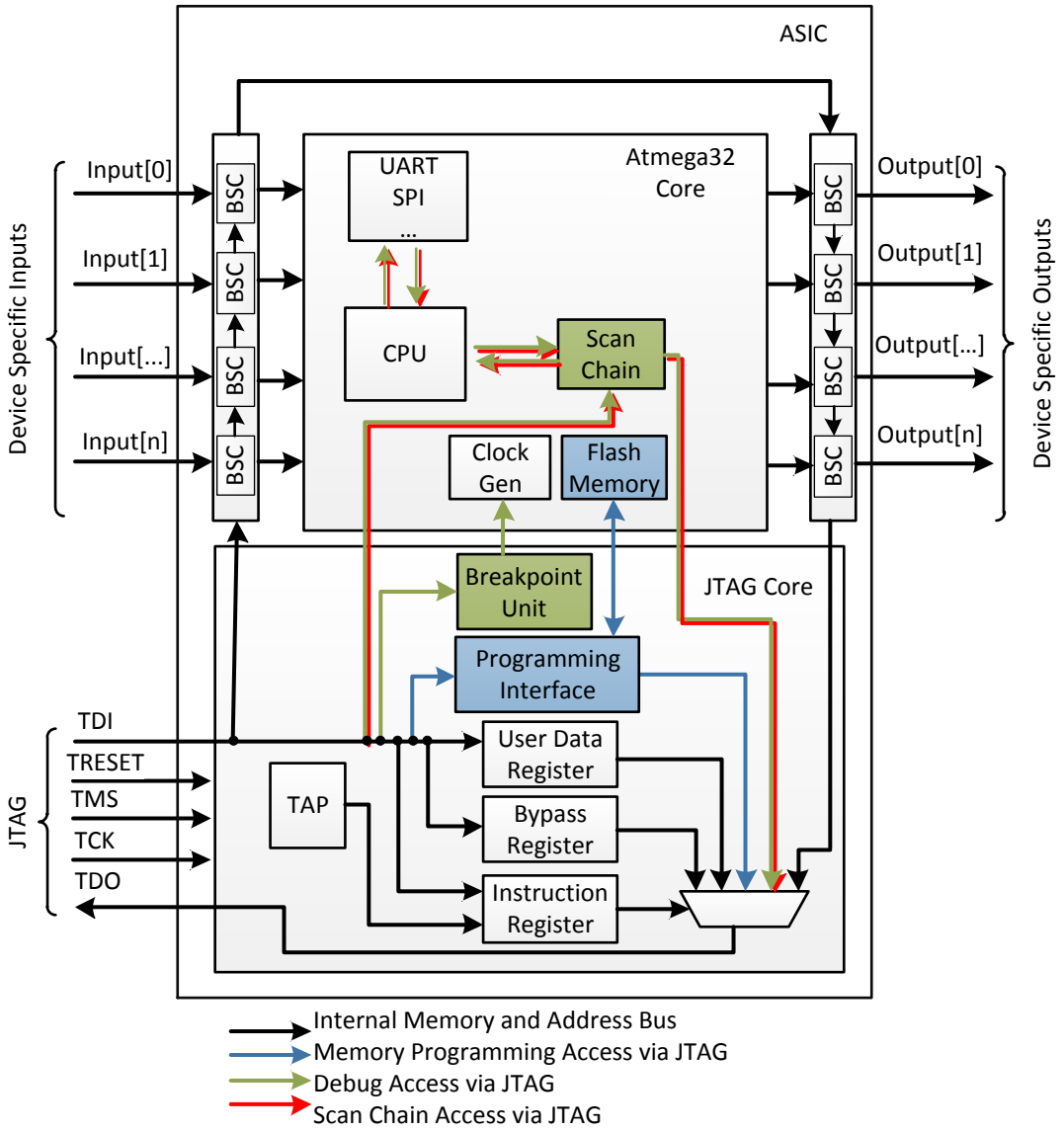


Figure 4: Microcontroller Atmega32 with JTAG interface. The JTAG allows an access to the internal scan chain, which is used for scan test (marked in green) as well as debugging (marked in red) as kind of a large shift register and even programming of the memory (marked in blue). Figure adapted from [3].

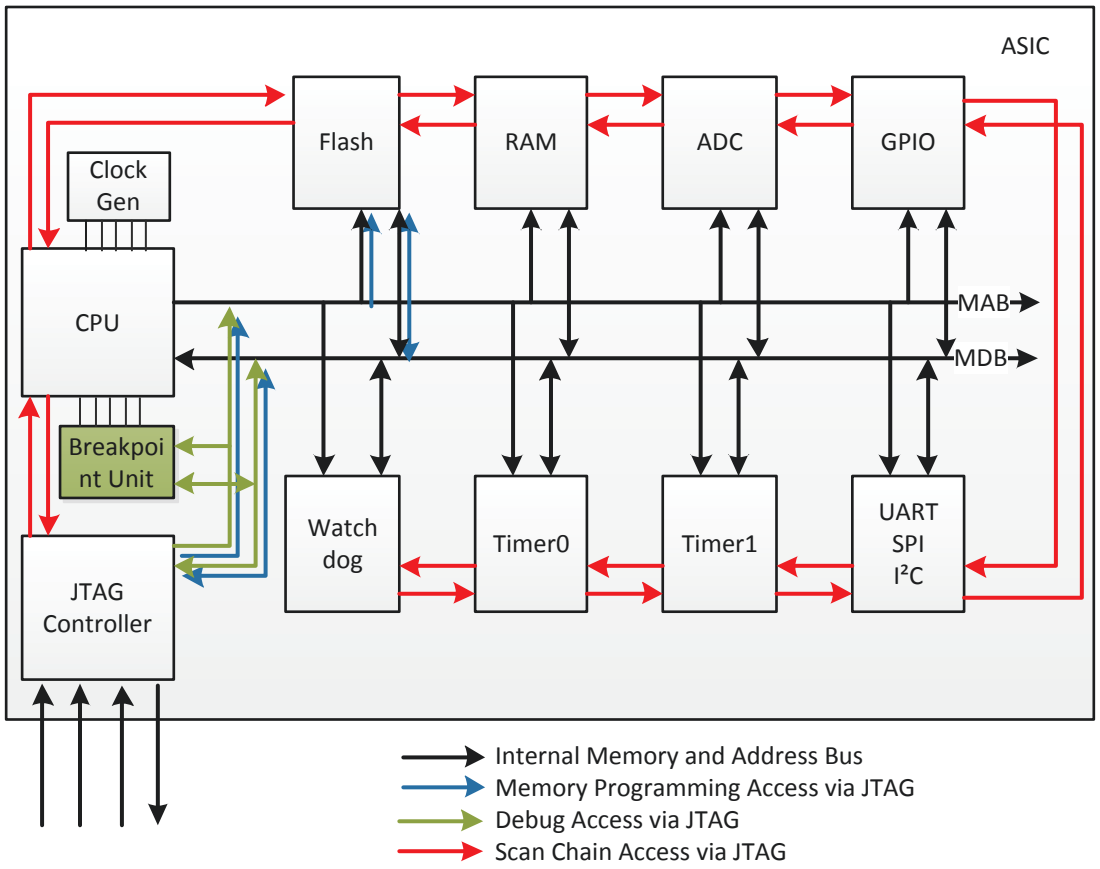


Figure 5: JTAG in MSP430 with direct access to memory data and address bus for debugging. The content of each address can be read or written individually, independent of anything. Figure adapted from [81].

It is also important to know the types of attacks, so that the developed security mechanism can be checked, whether it is vulnerable for attacks or not.

2.4.2 Read-back Attacks

Read-back attacks are very simple attacks. “Simple” in terms of the idea behind it and short in duration. Any unprotected device is vulnerable for such an attack without the high effort of a side channel attack. Dedicated test structures as well as programming interface allow an explicit access to most elements inside an ASIC.

In general the following attacks are possible:

- Extract data stored in registers, on RAM, ROM or flash
- Manipulate data, stored on a device
- Get information about the structure of the device to support side channel attacks
- Influence an active system

The first example is an attack against a microcontroller. Often a JTAG connection as pin header, or the port, is built with soldering pads and marked with corresponding signal names on the PCB. The programming and debugging interfaces are well documented in many cases and corresponding software to read out data is available from manufactures of ASICs. In a microcontroller, volatile memory for data storage and non-volatile memory for program storage exists. For some attackers the program itself might be in the focus of interest, for other attackers, it is to get access to accumulated data.

The second example is an attack against an FPGA. A verification feature can be misused by read-back attacks on FPGAs. After writing the configuration of the FPGA in a verification step, the configuration is read out and compared to the golden program. Such a read-back can be used to extract a hardware implementation or to get e.g. a secret key, which is part of the implementation.

2.4.3 Invasive Attacks

A physical attack means to analyze or to modify a device. Mainly for this type of attack, the device package needs to be opened. Since a modification of the ASIC is clearly visible, this type of attack is difficult to hide. Furthermore the risk of damage exists. For example the used acid, to open a package, can damage the die.

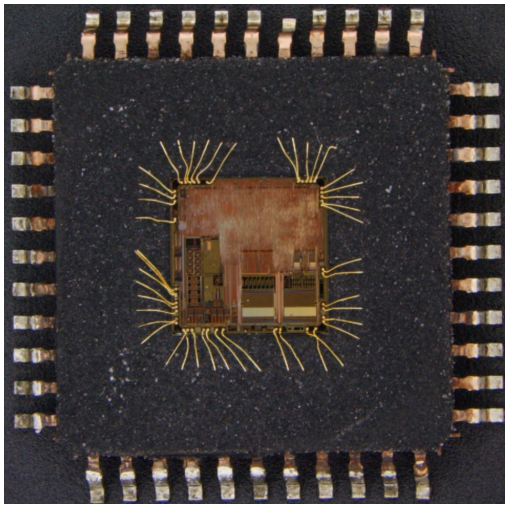
The most common physical attacks are:

- Read out the value of a laser fuse bit [26]
- Read out the value of an EEPROM [55]
- Repair fuse bit on a FIB [27]
- Probing attack - contact a wire with a small needle to get the signal value
- Laser attacks - influence content of memory elements
- Reverse engineering - destroying procedure to extract the schematic of a device

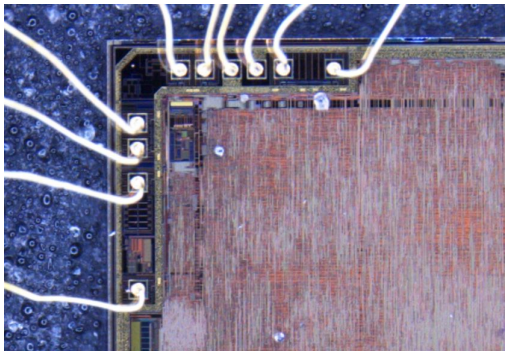
2.4.4 Package Opening

An ASIC in an open package can be helpful to attack the device. A carefully opened package, like the one in Figure 6, has intact bond wires, so that the device can be used under normal operation conditions. To get a fully functional device into an open package several experiments are required. Depending on the package type, plastic or epoxide resin, different types of acids have to be used [6].

At the end a small needle probe, which contacts a wire, can grab a signal or set the signal to a specific value into the device. Furthermore memories like flash [70] or SRAM [13] can be influenced by an attacker. Even an infrared camera can be used to identify highly active regions onto the die. If the attacker forces the usage of special functions, like encryption, a local heating is shown in an infrared picture. That can be helpful to reduce the effort in a reverse engineering step, because only a certain area has to be analyzed.



(a) Microcontroller in an opened package, ready to sold on a PCB for further attacks.



(b) Zoom in of Figure 6a in corner with intact bond wires.

Figure 6: Prepared microcontroller for microprobing or laser attack.

2.4.5 Reverse Engineering

"Reverse engineering" denotes the analyzing of a system. This could be a whole electronic system, like a video game console, or a PCB or even an ASIC. In the area of this thesis "reverse engineering" is a picture based recovering of the circuit. The motivation for an analysis can be different. At first it is a physical level analysis and the integrated circuit is inspected for production failures with a known schematic. The second motivation for a reverse engineering is the functional analysis. Such a functional analysis includes also the extraction of a secret key of a device.

The topic of "reverse engineering" is important especially in the area of WSN. In Figure 7 a commercial ASIC for sensor nodes is shown. All identified components are marked and labeled.

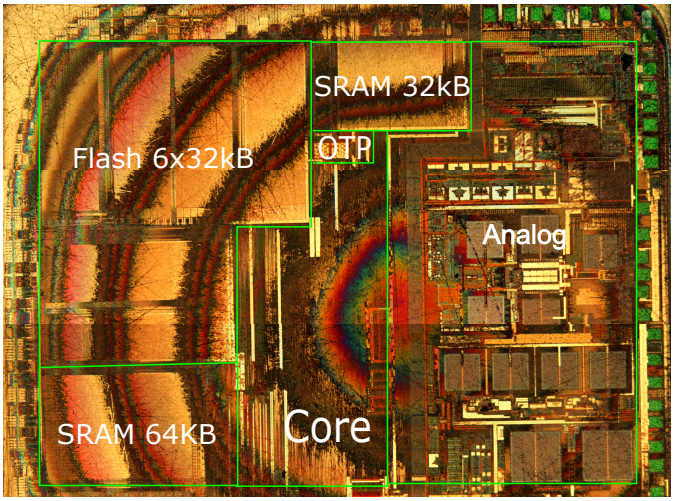


Figure 7: Microcontroller JN5139 after removing top metal layers. The ringlike structures is caused by the polish process. The outer material is removed faster than the inner one.

With external information such as the data sheet [44] and some experience, the core, all memory structures and the analogue components can be located.

Manufacturing technologies down to $180\,\mathrm{nm}$ can be analyzed using optical microscopes, which are used for different microcontrollers. These microscopes are available in many laboratories or they are cheap to buy. So in principle any attacker is able to extract information from a Device under Attack (DUA). The quality of a polished ASIC and of the photographs depends on the budget and the experience of the attacker. A detailed tutorial for reverse engineering can be found in [47]. The general idea of the reverse engineering of analogue and digital circuits, is an iterative process of photography, followed by removing the current metallization layer, followed by a next photography and so until photographs of all metallization layers are taken.

In Figure 8 a vertical cut through a silicon technology is shown. It can be seen, that metallization layer and each vertical connection (via) has a small high of less than $1\,\mu\mathrm{m}$. So, the removing a layer is a very difficult task, because only a very small amount of material has to be removed to be successful. In Figure 9 a small cutout of a die with four different metal layers is shown. In the first one (Figure 9a) all metallization layer are visible, in the second one (Figure 9b) several layers are removed so that the vias from metal 2 down to metal 1 are visible. Although the diameter of the vias is less than $360\,\mathrm{nm}$, the vias are clearly visible. In the last Figure 9c only the structures on metal 1 are visible. To generate the netlist of the reviewed device, the function of the digital standard cells is required. Finally the wiring of all cells is extracted from photographs of each metallization layer. Tools like degate [46] supports this complex task.

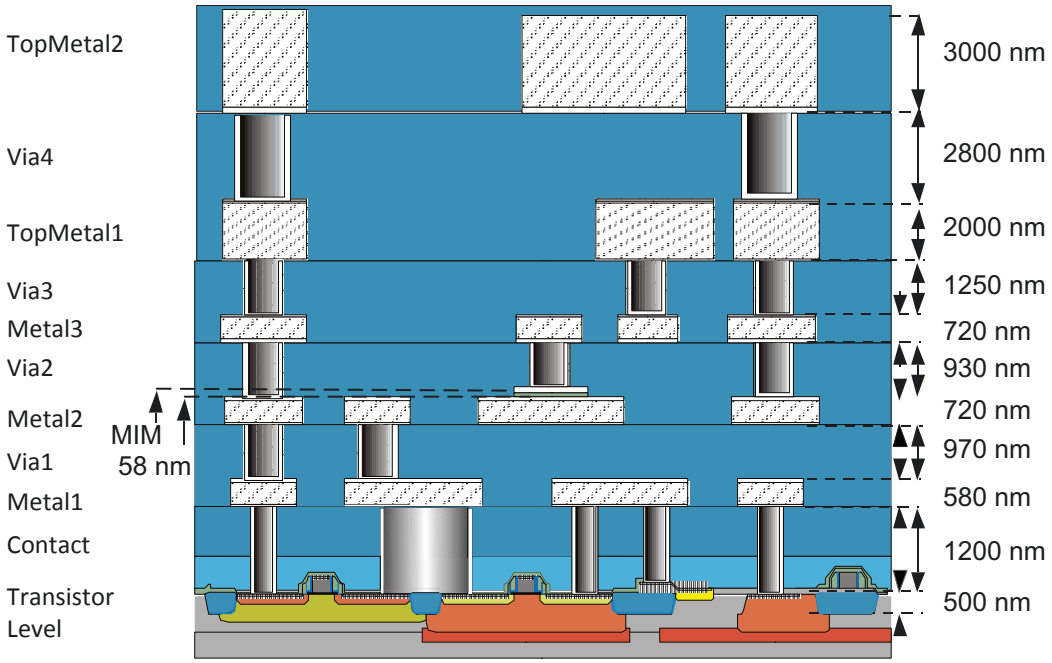


Figure 8: Cross section of IHP $0.25\,\mu m$ technology with dimensions of metallization layers and vias.

2.5 Memory Types

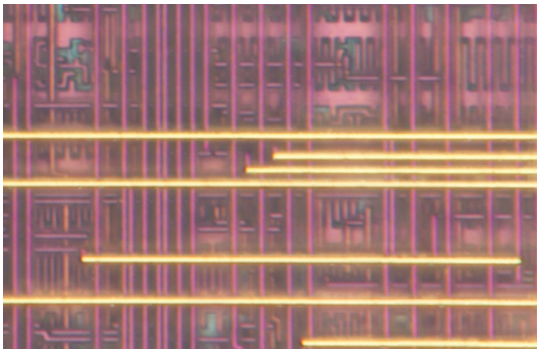
In this section different memory types are presented. At first, non-volatile memory is required to store the golden key. Second, the secret key has to be stored at least for the time window which is required to perform the post-production test or the debugging session.

Depending on the application area, a memory should be writable only once or more than once. In the case of a device ID, the value should not be changeable, so an OTP is a suitable solution. If the memory contains an application program, reprogramming can be useful to update software, e.g. to eliminate bugs.

One requirement for the solution proposed in this thesis is, that the key is stored at least for the time to test or debug. The key, which is written by the user to enable the corresponding function (secure scan chain or debug interface), can be stored in a volatile memory. This has the advantage that a security leak does not occur in case of the user forgetting to delete the secret key after usage. Such a memory can be implemented as SRAM or can be flip flop based. These types of volatile memory are very common. Any other type of memory like DRAM is not well suited to the area of sensor nodes.

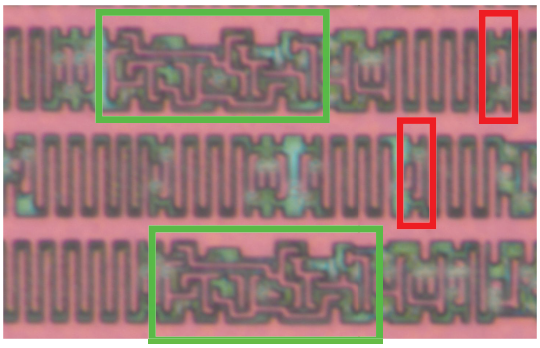
2.5.1 Volatile Memory

Flip Flops Flip flops are standard cells in a digital library and they are placed in rows in the layout as any other digital standard cell. A schematic of flip flop is shown in Figure 10a and



(a) Detail of an ASIC with three standard cell rows and a wiring up to metal layer 4. Two horizontal and two vertical metal layers are visible.

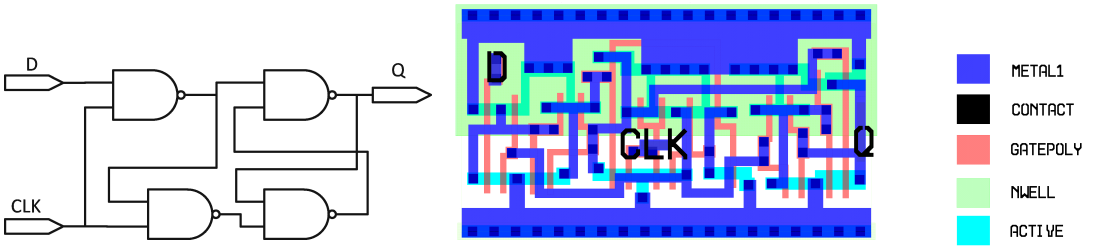
(b) Photograph of a polished ASIC. The higher metal layer are removed and now three standard cell rows with metal layer 1 and vias to metal layer 2 are visible.



(c) Photograph of a polished ASIC down to metal layer 1. The flip flops are marked in green and inverters in red as example for a practical reverse engineering.

Figure 9: Photography of an ASIC with different number of metallization layers in IHP $0.25\,\mu\mathrm{m}$ technology, from top to bottom. This is a usual view during a reverse engineering with removing of each metallization layer by polishing.

the corresponding layout in Figure 10b. Figure 9c shows a small section of an ASIC layout, in which a flip flop is marked in green. In principle, a flip flop is easily identifiable in optical inspection, if less metal layers are used, because it is one of the largest cell types of a library. In case of three or more metal layers for the wiring, it is more complicated to identify the different cell types without mechanical polishing. E.g. in Figure 9a an example is given, in which all cells are covered with wires. Although only four metal layers are used, the cell types are hard to identify.



(a) Schematic of a flip flop, based on NAND gates in feedback loop.

(b) Layout from schematic 10a in IHP 0.25 µm technology. An optimized layout causes that the NAND gates in feedback loop cannot be identified easily.

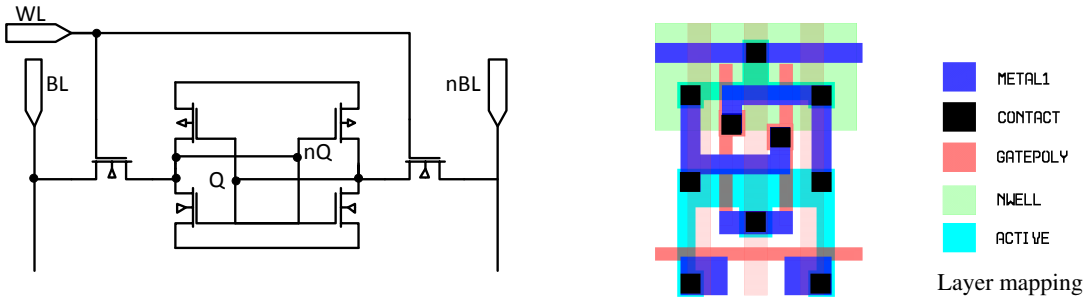
Figure 10: Schematic and layout of a flip flop

SRAM This type of memory is widely used in many application areas and is available in most digital design kits. The data is written into specifically designed cells and the value is stored until power is switched off. In inactive mode, which means that no input signals is changed, SRAM has very low power consumption.

One usual SRAM cell requires six transistors to store one bit value (Figure 11a), additionally an address multiplexer for controlling and amplifiers for read and write operations are needed. For very small memories an SRAM is not well suited, because the additional components increase the required area. Especially if the data word has an increased width (more than 16 bit), the required area for amplifiers becomes significant, as each bit requires an amplifier. In the end, an SRAM module is well identifiable in layout, which could be helpful in a reverse engineering step or in a probing attack.

2.5.2 Non-volatile Memory

In a non-volatile memory the value is retained, even if the power supply is interrupted. So in this section different types of non-volatile memories are described. First, different kind of memories are shown, whose content must be defined before production of the ASIC. Second, types of memories, which can be programmed after production, and third, memories, which can be re-written several times, are explained.



(a) Schematic of SRAM. Two cross coupled inverters and two access transistors.

(b) SRAM layout.

Figure 11: Schematic and example layout of six transistor SRAM cell

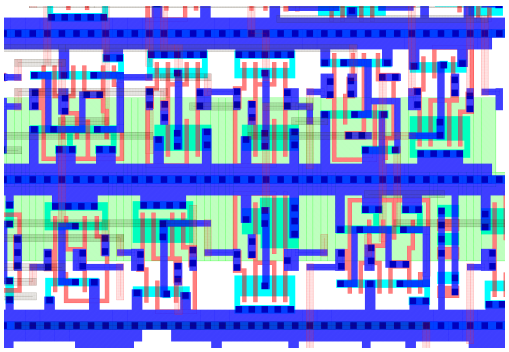
Combinational Logic ROM A very simple ROM is based on combinational logic cells. A look-up table, for each input value, has a specific output value and is built up by digital standard cells. A possible implementation in VHDL is shown in listing 1. The Figure 12a is a photograph of a ROM, which is made out of digital standards cells. The cells are distributed in a common area for all digital standard cells of the ASIC and it is not possible to identify them without significant effort. For an attacker it is unclear, which combinational cells are part of the ROM and which cells are part of the e.g. microcontroller.

Listing 1: Combinational ROM in VHDL

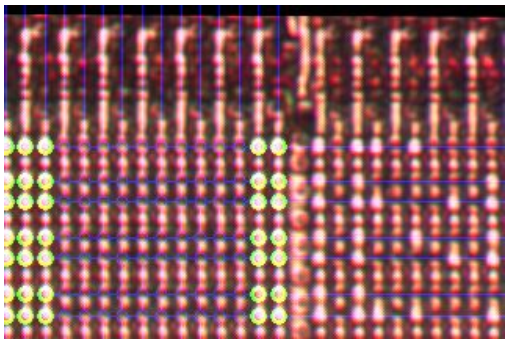
```
case input is
  when x"01" => output <= x"63";
  when x"02" => output <= x"7c";
  when x"03" => output <= x"77";
  when x"04" => output <= x"7B";
  ...
  when x"ff" => output <= x"16";
end case;
```

Array ROM In Figure 12b a ROM array, called “masked ROM”, is shown. The horizontal wires are “word lines” and the vertical wires “bit lines”. The content, which means every stored bit, is programmed at the cross point of word and bit lines by placing active material. As for the combinational ROM, the programming of array ROM is fixed for all ASICs with one mask set. If the content should be changed, a new set of masks and a new device production is necessary, which is expensive and time consuming. In case the ROM is used as program memory, software development process must be finished before the ASIC gets produced.

Laser fuse OTP For laser fuses, a laser beam is used to cut a short stripe of polysilicon or metal with a width of 2 µm to 3 µm. The cell size is very large and it is only usable for ASICs, which are on a wafer or in a package with a window, but not in closed packages. The typical application area is for memory repair. A defect memory area is replaced by a spare module.



(a) Combinational ROM in layout view in IHP 0.25 µm technology.

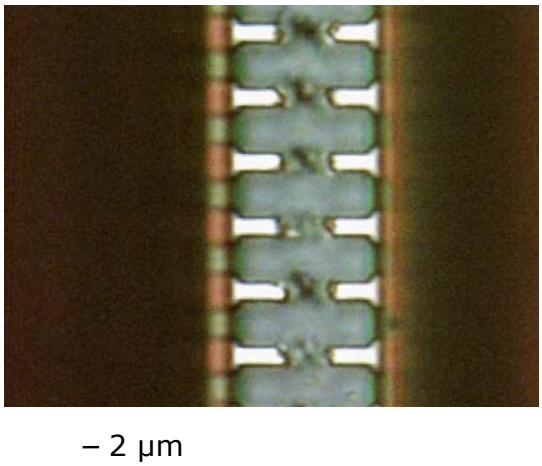


(b) Array ROM in layout. Some "ones" are marked with a green circle. Figure reprinted from [1].

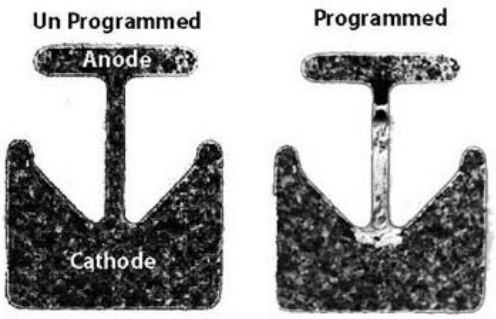
Figure 12: Different types of ROM

Figure 13a shows laser fuses which are already cut. Contrary to other components in CMOS technology, a laser fuse cannot be build smaller than 2 µm to 3 µm. A very small laser fuse cannot be cut, because the laser spot size is to large. Reducing the size of the laser spot is a future challenge.

Electrical fuse OTP An electrical fuse is a storage element, which is fully compatible to the CMOS process. Such a fuse is made of polysilicon (see Figure 13b). The cells are programmed one by one to keep the required total of the current at 20 mA. In the end the former electrical connection is cut. Similar to a laser fuse, the required area is relatively large and the value can be read out by optical inspection as seen in Figure 13b. A detailed comparison to all fuse types is given in [21].



(a) Cut laser fuse. Figure reprinted from [26].



(b) Electrical fuse OTP. Figure reprinted from [14].

Figure 13: Laser and electrical fuse OTP

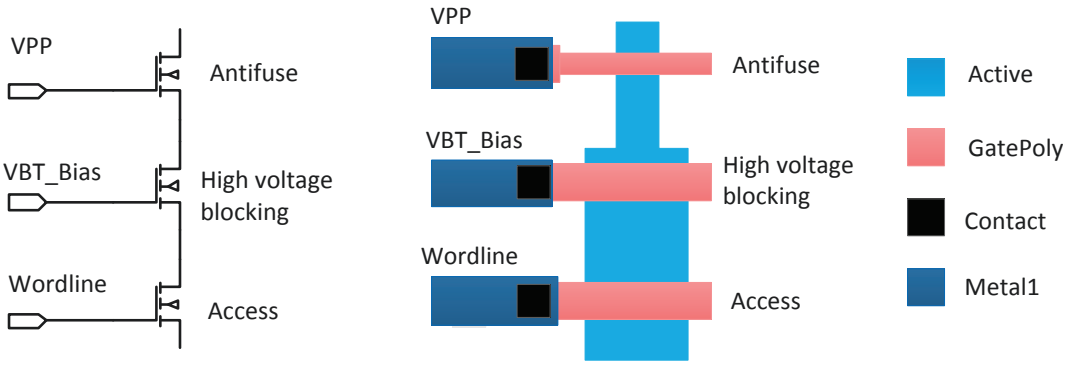
Anti-fuse OTP While a fuse has a low resistance (a conductive connection as initial value) the anti-fuse has a high resistance (non-conductive connection) at the beginning. An anti-fuse bit cell is based on a transistor (Figure 14), in which the gate oxide is the anti-fuse element. A high voltage between gate and source causes a "hard gate oxide breakdown" and the isolated connection becomes a conducting connection. Before the breakdown, the gate resistance is very high, around 1 GΩ. After the programming phase, the resistance is less than 100 kΩ. During programming of a single bit cell does not require additional components, the reading process requires a sense amplifier.

Like the electrical fuse OTP, the anti-fuse OTP cells can be fabricated in a standard CMOS process without additional steps [35]. An nmos as well as a pmos transistor can be the basis of the OTP cell. The high voltage is only required to program the fuse bit, so a thick gate-oxide for transistors is not necessary. The advantage is that the design principle is usable even for very small technologies, in which a thick gate-oxide is not available.

In the layout, the cells are clearly identifiable. The differences between an anti-fuse OTP cell with sense amplifier and a digital standard cell are significant. The three transistors with a common active area would be an unusual structure for digital standard cells.

Another implementation type of an anti-fuse OTP is a metal-to-metal anti-fuse [90] shown in Figure 15. Between two different wires on two different metal layers a via and an isolator are placed. In case of a high voltage, the isolator will breakdown and the connection between the different wires is established. In [69] this type of anti-fuse is classified under "extremely high security level".

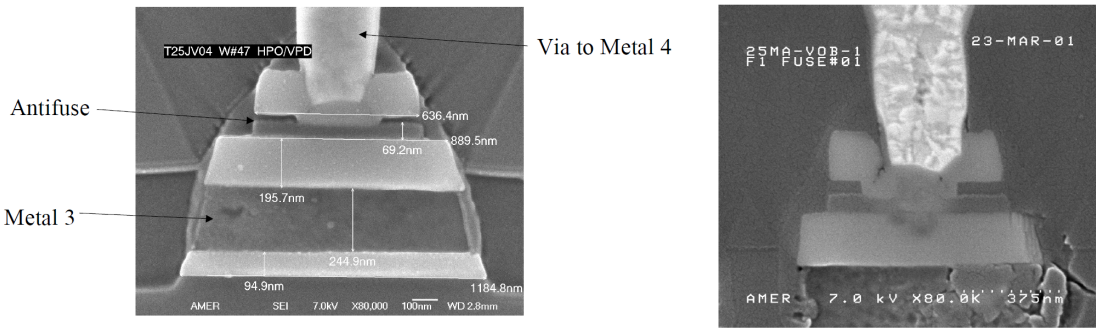
No matter in which variants anti-fuse cells are implemented, it is impossible to read out the value via optical inspection, by using a microscope. At least a FIB is required because the anti-fuse area is so small that it cannot be seen optically. In Figure 15 an example of such an analysis is shown.



(a) Schematic of single OTP cell

(b) Layout of single OTP cell. The anti-fuse transistor is much smaller than the other ones.

Figure 14: Example for transistor-based anti-fuse OTP.



(a) Unprogrammed anti-fuse cell

(b) Programmed anti-fuse cell

Figure 15: Metal-base anti-fuse OTP in TEM photography. Figure reprinted from [50].

EPROM The content of an EPROM can be programmed after production and furthermore it can be erased [23]. The disadvantage of a ROM, programming during production phase, is solved by use of this memory type.

Each cell has two gates, which are on top of each other as shown in Figure 16. To store a value, a high voltage is applied at the top gate electrode for the corresponding bit cell. The high electric potential causes a placement of electrons between both gates and the effect is called Hot-Carrier Injection (HCI). The programmed value, the injected electrons, can only be erased for the whole memory array, by UV light. For such a circuit, a special package with a small quartz window is required. If the memory is embedded into a standard plastic package without a window, the memory can be used as OTP. While a ROM can be fabricated without any additional costs, the EPROM requires additional masks to fabricate an additional gate on top of the other one.

Flash Memory Flash memory is a type of EPROM which can be erased, too. In contrast to EPROM the erase process is done in an electrical way, so that no special package is required. To perform the erase process, a voltage is applied to the floating gate (see Figure 16). Additional circuits like an internal voltage regulator or transistors to erase the content of the memory are necessary.

RRAM In previous years, several new memory techniques were developed, such as FeRAM, MRAM or RRAM. An example of the new generation of fast NVM types, the resistive RAM, is shown.

In Figure 17a the schematic of the RRAM is shown. The basis of this memory type is a Metal-Insulator-Metal-system Figure (17b), which is enhanced by an HfO_2 layer. In principle, a via connects metal structures on two different metal layers. The HfO_2 layer is placed under the via and this works as resistor. An electroforming process changes the value of the resistor. A low resistance causes the logical value “1” and a high resistance the logical value “0”. An nmos transistor is used to read out the value.

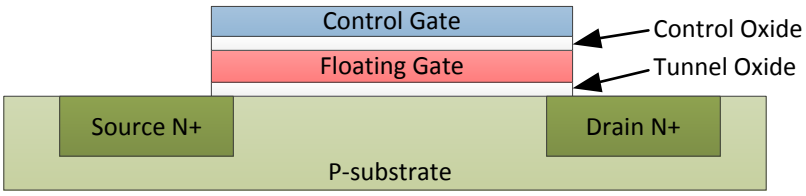
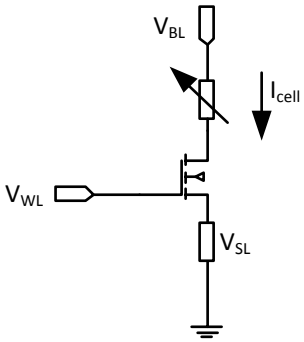
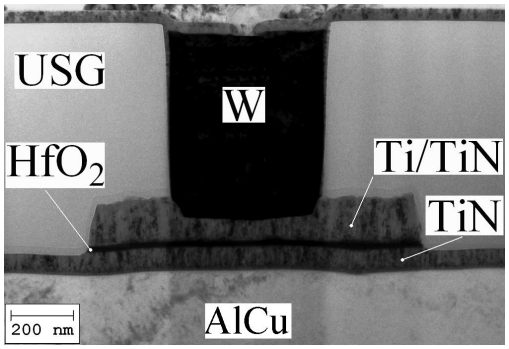


Figure 16: Design principle of flash cell in vertical view. The double gate implements the store functionality. In the floating gate are injected electrons. The structure of flash and EPROM cell is the same; the different is in the additional module for the erase process.

In [20] the advantages of RRAM as secure memory are described. The content cannot be read out optically, nor can it be influenced by a backside attack, using a laser.



(a) Schematic of RRAM cell



(b) TEM photography of RRAM cell in IHP 0.25 µm technology. On top is Metal3. A via, made of wolfram, goes in direction of Metal2. The direct connection between Metal2 and Metal3 is interrupted by a small hafnium oxide layer. The Hafnium oxide is responsible for resistive effect.

Figure 17: Schematic and implementation of RRAM cell.

PUF A very special non-volatile memory is a Physical Unclonable Function (PUF). Such a PUF cannot store a user defined value. It generates an individual identifier or secret key, e.g. 64 or 128 bit, for every device based on deviations in the production process. The goal of a PUF is to generate a value which always is the same for a specific device at any time, independent of operation conditions like temperature or voltage. Since the value of a PUF cannot be influenced during fabrication, the value is individual to every device.

Well known types of PUFs are the delay PUF and the SRAM PUF. One bit of the delay PUF requires two identically implemented delay paths. In a race condition it is decided whether the corresponding value is “0” or “1”. For an SRAM based PUF the initial value of the memory is used. A time consuming post-processing, e.g. with support of BCH [45], of the initial value, eliminates single bit faults.

2.6 Camouflaging Structures

“Camouflaging” in the context of ASICs means a type of security mechanism, which is able to increase the effort for the reverse engineering. Additional structures without influence on the behavior of the design are examples for such techniques. The effort during the reverse engineering is increased for the attacker and it is to be assumed that the error rate is increased or it is completely impossible to extract the netlist.

Another approach for a camouflaging technique is the use of a conductive or non-conductive connection, which is not visible in optical inspection. In a microscope the attacker sees only a flat view in 2D.

In the following camouflaging structures at different levels are explained. They are the basis for the designed compare units CMP0 and CMP1 in this thesis. These units are a main feature of this thesis.

2.6.1 Transistor Level

In the first step during ASIC fabrication the nmos and pmos transistor are produced. So the first element of all layers in the ASIC production is the transistor. Even this device can be modified in its functionality without changing the visual appearance. In [4] a permanently on transistor is shown, which is helpful to confuse an attacker in a reverse engineering step. In an analogue layout the transistor can be used to implement a circuit, which is always connected to power on. An attacker would identify the transistor as power on and power off switching element. For digital circuits the permanently on transistor can be used to implement logic gates with another function as it looks like.

2.6.2 Logic Gate Level

In a standard digital library each cell has a specific layout as shown in Figure 18a. An experienced designer is able to identify the functionality of each cell; this is required for successful reverse engineering. In Figure 18b the optical view is shown, which means this is what in an optical inspection can be seen. The appearance of the NOR and NAND cell differs clearly.

A countermeasure against reverse engineering is a digital standard cell library; within all cells have a common layout, independent of its logical function. An example for such a library is given in [5]. The NAND and NOR cells have the same layout in optical inspection. The different between both cell types is the placement of implants at different positions. The implant is used to build a conductive connection at a very low layer. It is not visible in optical techniques, so that the function of the standard cell cannot be determined. In Figure 18c a layout of NOR and NAND cell is shown. The n-well layer is marked in green and the active layer is marked

in turquoise. Except of these two layers, both cells have an identical layout. Even in reverse engineering with removing every metal layer, the n-well and active layer cannot be seen. So an attacker gets the image, which is shown in Figure 18d.

The idea of the dummy contact is presented in [12]. It allows another camouflaging approach. Between a transistor and the wiring metal layer a contact or a dummy contact is placed. In an optical inspection is no difference between the conductive and non-conductive connection. A very careful reverse engineering step is required to differentiate both types. In [62] the idea is used, to implement a standard cell library. In a pure optical inspection the function of the cell cannot be determined.

An advantage of such modified digital libraries is that these libraries are available from foundries or specialized companies [43]. With such a library the design flow is equal to the design flow for developing an "unsecure" ASIC[1], the library is verified and support is given.

A disadvantage of this approach is the need to redesign of all cells into a design which looks equal for all cell types. So the cell design is not specialized to its function anymore. This causes an increased area consumption, power consumption and internal cell delay. In [62] a factor of 4 to 5 is given for area and power consumption. The delay is getting worse up to a factor of 1.6. Furthermore it can be clearly seen, that special cells are used in a design and an attacker might be more interested into the ASIC.

2.6.3 Wiring Level and Coating

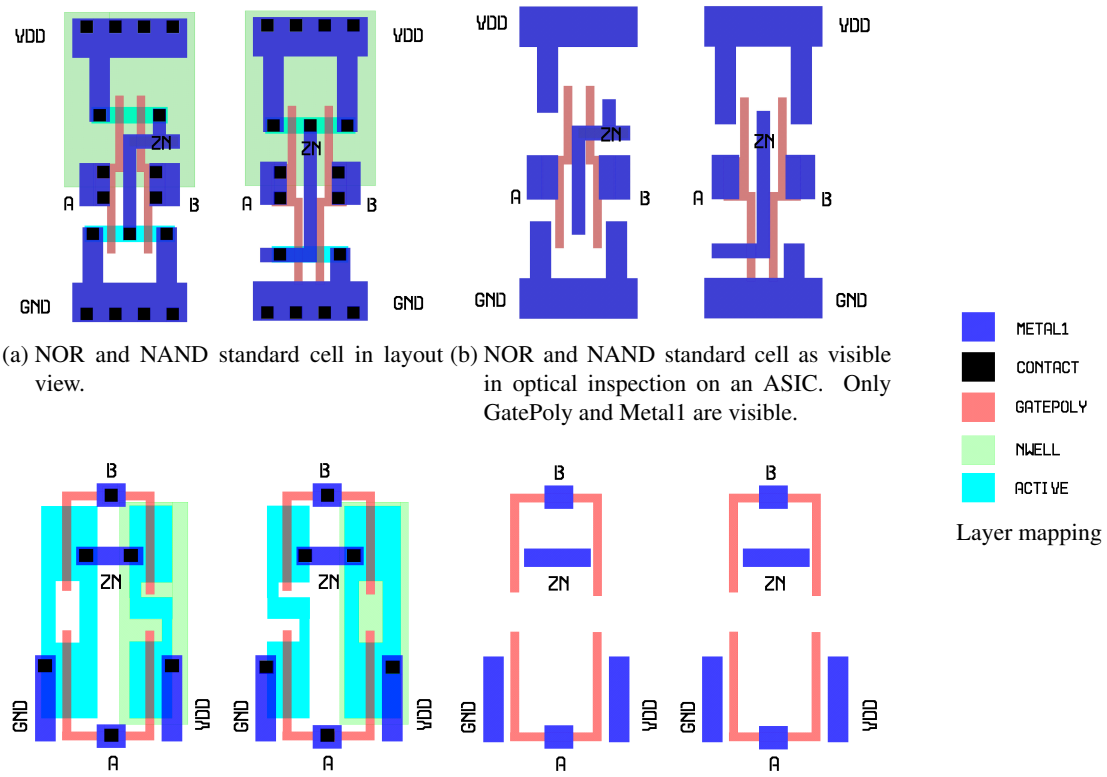
A very simple type of protection mechanism against reverse engineering is the hiding of structures under wide metal stripes. This countermeasure works only in cases in which the attacker does not remove each metal layer in polishing steps. In Figure 19 an example is given, in which the wide stripe on metal 2 covers a part of the standard cells. In a visual inspection, it is not clear, which cells are inverters and which cells are filler cells. As result it is unclear for an attacker in Figure 19, whether the signal is inverted or just buffered.

A further protection mechanism is "coating". It means that the whole die is covered by a metal layer without any function for the signal routing, but it can be used for distributing the power supply. In Figure 20a a microcontroller without a top metallization is shown. So, in Figure 20b details of the digital standard cells can be seen in an optical inspection. The microcontroller in Figure 20c has a full top metallization and details in metal layers below cannot be seen. Furthermore the top metal coating prevents simple laser attacks, in which the content of a memory is influenced by a laser beam.

2.6.4 Conclusion

"Camouflaging" is a common approach to make the reverse engineering of an ASIC harder. Some techniques like coating do not influence the power consumption, but digital standard cells with a common layout influence the power consumption significantly, so that this approach is not applicable in the area of battery powered devices. Nevertheless the high number of patents in this area shows, that there is a big market.

[1] An "unsecure" ASIC means a design which does not include countermeasures against reverse engineering.



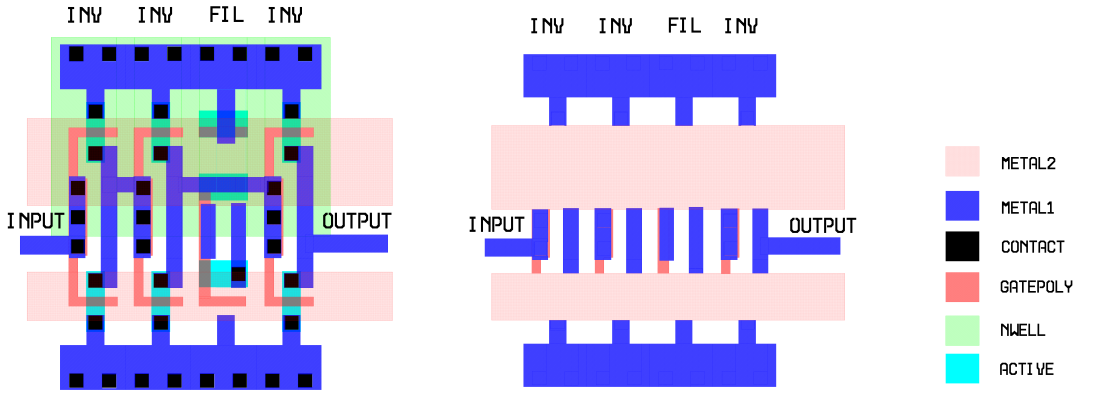
(a) NOR and NAND standard cell in layout view.

(b) NOR and NAND standard cell as visible in optical inspection on an ASIC. Only GatePoly and Metal1 are visible.

(c) NOR and NAND cell in camouflage technique. The layout of both cells is similar, the difference in the functionality is made by different active regions.

(d) NOR and NAND cell in camouflage technique in optical inspection on an ASIC. Only GatePoly and Metal1 are visible, so that the cells have the same appearance.

Figure 18: Layout of NOR and NAND cell from IHP $0.25\,\mu\mathrm{m}$ standard cell library (Figure a and b) and implemented in camouflage technique (Figure c and d). Figure c and d reprinted from [5].

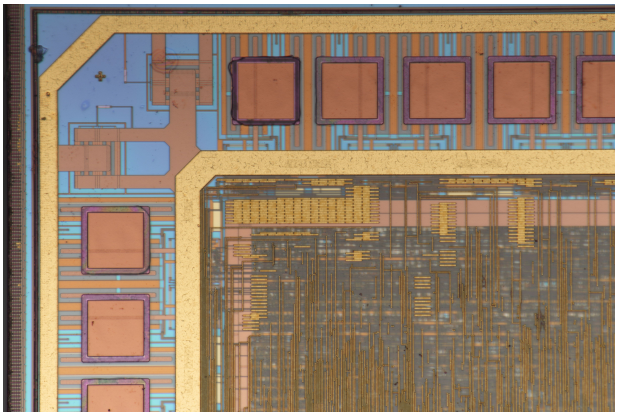


(a) Placed digital standard cells, three inverters, one filler cell in the layout view.

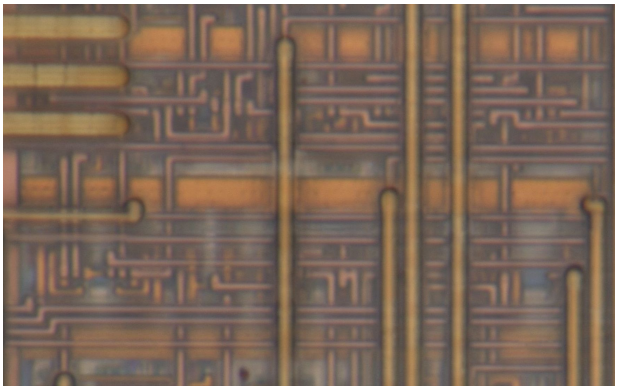
(b) Circuit as it looks like in optical inspection on an ASIC. The wide Metal2 wire covers the cells, so that the schematic of the circuit is not clearly determinable.

Layer mapping

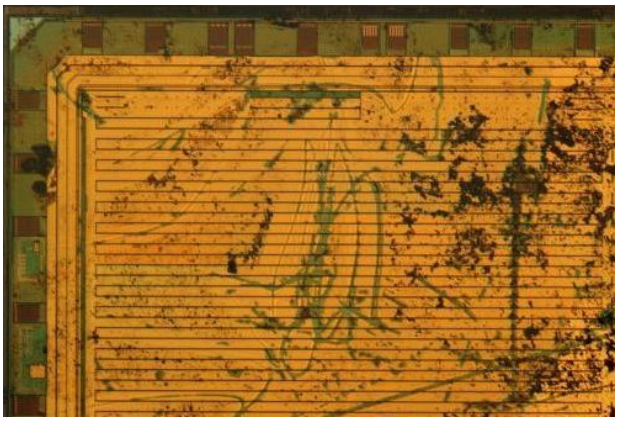
Figure 19: Simple camouflage technique at wiring level with IHP $0.25\,\mu\mathrm{m}$ standard cell library. Without removing of metal layers, the type of each cell cannot be identified clearly.



(a) Photography of a corner of a microcontroller. Bond pad area (upper and left side) and area with digital standard cells and their wiring is visible.



(b) Zoom of Figure 20a, so that two standard cell rows and a wiring on Metal layer 2 and 3 are visible.



(c) Photography of a microcontroller. Bond pads and wide metal stripes are visible only.

Figure 20: ASIC without and with top metal coating.

3 State of the Art – Test and Debug

In the following section an overview of the post production test of ASICs, possible attacks and countermeasures is given. Later in this thesis the secured scan interface are checked against these attacks and the weak points of the countermeasures are analyzed.

3.1 Test

The post-production test verifies, that the produced devices fulfill a given specification and, typically, it includes two types of tests. On one hand is the parametric electrical test. Already in this test, some devices may fail due to shorts and opens of the pads or the leakage current is larger than expected. On the other hand the device test is the functional verification of the logic which covers most defects in transistors and connections. This is a very complex task compared to the parametric test. For an extensive functional test after production controllability as well as observability are a precondition.

Typically a simple functional test of a complex ASIC does not have a high test coverage and requires a long test time in most cases. Such a test would be very slow and furthermore an error analysis is very complicated. E.g. a pure functional test of a counter with a size of 32 bit requires 4 billion clock cycles to verify that every bit can flip from “0” to “1”. Based on this lack of a very time consuming functional test, explicit post-production tests are developed in 1960’s. To support the post-production test two different approaches are very common. The first one is the scan test [37], and the second one is an exhaustive self-test, called Built-In Self-Test (BIST) [49].

3.1.1 Test vs. Security

Quality requirements of high-volume ASIC production can be fulfilled by post-production tests only. Due to the fact that every ASIC has up to several million transistors, production errors cannot be avoided. To ensure that the delivered devices are fully correct, a comprehensive test of each device has to be done. From the economic point of view it is important to exclude defect devices from further processing, to avoid additional costs.

To enable an efficient test, a scan chain is built into an ASIC in most cases. Such a scan chain is easy to integrate and the test coverage is very high. Furthermore, a defect analysis is possible and it can be used to optimize the production process and to increase the yield. The yield rate can be improved by analyzing productions errors. For such an analysis, it has to be clearly identified at which gate an error occurs more often. The design and input and output test pattern are put into a software. The software compares the expected output and the received ones and calculates the gate, which is responsible for the error.

The requirements for an extensive test and fault analyzing are contradicting to a secure design. On one hand, an extensive test requires full access to any module in the device. On the other hand, a secure design does not leak any information. Security leakages occur, if the scan chain contains secret data, like a private key, or intermediate results, which can be used to retrieve the secret data.

A BIST compresses the output data to a minimum value. In its extreme form, only two output signals, BIST done and BIST fail, are generated. This is a secure solution, but it does not give any idea about any error. The only information provided is that at least one error has occurred during self-test. It means that a BIST is not an option.

3.1.2 Scan Test

A well-known technique to support an efficient test is the insertion of a scan chain. The scan test technology was introduced by Kobayashi in [37]. The design principle is the access to the storage elements, especially flip flops, inside of ASIC with a minimum number of additional pins. As result all flip flops are combined to a long shift register with external pins for input and output. This shift register is called "scan chain". An example for scan chain insertion is given in Figure 21a and 21b. The first figure shows a circuit after synthesis without scan flip flops, the second figure shows a circuit, in which the combinatorial logic is unchanged, but all flip flops are replaced by scan flip flops. To test a device this scan chain is used to shift in a predefined pattern, apply one working clock cycle and compare the output with computed results.

A well testable design has test coverage of more than $99\,\%$ for stuck-at faults. To reach such a value, almost all flip flops of the design have to be replaced by scan flip flops and they become part of a scan chain. Since a scan chain is a large shift register the device has a high controllability because the input of each register can be set individually, independent of other flip flops. The observability is very high as well; because of output of the large shift register is connected to an output pad of the device.

This technique can be used to identify different types of faults like stuck-at and delay faults or opens and shorts. In the corresponding test pattern generator the fault models are set and the required test patterns are calculated. Since the functional test of a 32 bit counter, requires several billion clock cycles, an exhaustive post-production test using a scan chain is much faster. The example design requires $2{,}475$ clock cycles only to test the device on stack-at faults.

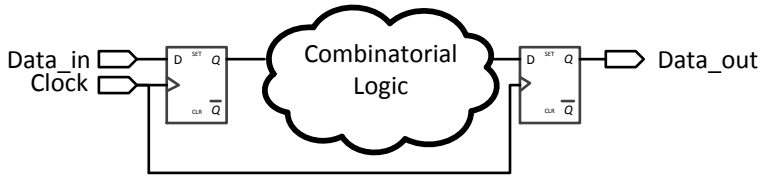
Stuck-at faults can be found by a simple test vector, but delay faults are more complicated to identify. Delay faults and their test require more than one test vector and it has to be executed at the same clock frequency as the one for which the ASIC was designed.

Integrating a scan chain into an ASIC is a straight-forward task for a standard design. The input and output pins can be shared with functional pins of the device. Only the `scan_enable` pin has to be a separate input pin. Furthermore, it is possible to combine the scan chain with an JTAG interface as shown in Section 2.3.

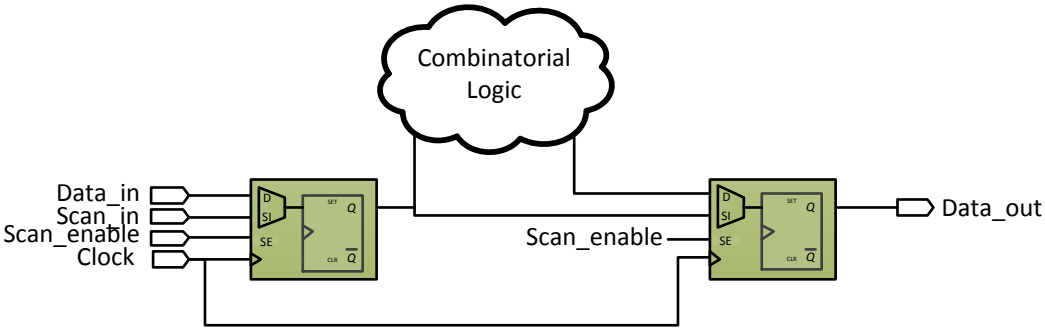
Many tools like TetraMax (Synopsys) [78] or Encounter DFT Architect (Cadence) [9] support the scan insertion and test pattern generation for the given fault models. Flip flops within a design are replaced by "scan flip flops" as shown in Figure 21 but the logic is untouched. Only the type of the flip flops is changed, which is a well applicable task. Figure 21a shows the differences between a normal flip flop and scan flip flop in Figure 21b. A standard flip flop is extended by a multiplexer at the data input. The `scan_enable` signal controls the multiplexer and switches the design from functional mode to scan mode and vice versa. In functional mode the flip flop gets its data from the combinatorial logic. If the `scan_enable` signal is asserted, the scan flip flop receives its data from output of preceding scan flip flop. The timing of the `scan_enable` signal is uncritical so that many flip flops are controllable without a large buffer tree.

The additional multiplexer at the input influences the timing of the storage element. This is the only point where the timing is getting worse during the scan insertion.

In some cases a flip flop cannot be integrated into a scan chain. For example, inside of a clock controller with clock divider because the `clock` signal is fed through this component and the `clock` signal has to be stable for the while duration of the test. So, the clock controller is tested by a small functional test, e.g. setting different clock frequencies and a verification step in which the adjusted frequency is observed.



(a) Flip flops and combinatorial logic.



(b) Scan flip flop and combinatorial logic

Figure 21: Design without and with scan flip flops

The following sequence is illustrated in Figure 22, which has to be executed to perform a scan test. The execution of the scan test is done in the following way:

1. Set scan enable signal
2. Apply scan input data and one clock cycle for each data value
3. Release scan enable signal
4. Apply one clock cycle
5. Set scan enable signal
6. Apply a clock cycle for each data value / flip flop and read the output

All flip flops in a scan chain can be filled with any data. Afterwards one working clock cycle is executed, which is the shortest execution window in an ASIC. At the end the content of any flip flop can be read out. The difference in the data, before and after applying the clock cycle,

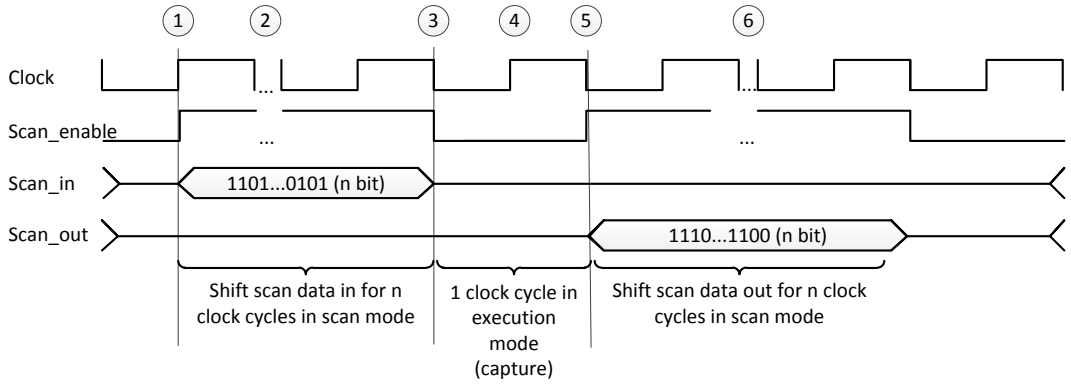


Figure 22: Waveform of scan test. The number of clock cycles n is equal to the number of flip flops in the scan chain.

allows to draw conclusions about the combinational functions. The design is full controllable and it can be analyzed, independent of a test engineer or an attacker performs the access.

3.1.3 Built-In Self-Test

A BIST is used for the self-test of a device. Two different approaches are known for such a BIST, on one hand the offline [49] and on the other hand the online test [10]. In an offline test a special test mode is entered and the test is executed. This means that in this time the device is not in operation mode.

A device test during operation mode is called “online BIST”. This self-checking mechanism can be divided into two different categories. In concurrent mode a duplication and comparison is performed during the operation. In a non-concurrent online test the idle time of a device is used to test the device. Special diagnostic routines in software are executed, but they can be interrupted at any time to switch back in normal operation mode.

To implement an offline BIST functionality of an ASIC, four additional circuits have to be built into the device. In Figure 23 an approach for an offline test is shown. The JTAG interface can be used to set the BIST configuration inside the BIST controller, starting and read of the result. In the pseudorandom number generator (PRNG) the input data for the internal scan chains is generated. A Multiple-Input Signature Register (MISR) is used as compactor module and compare unit and it checks the result with the expected value.

As an advantage it can be seen that a BIST is a test with a low effort during the test procedure. It is not necessary to apply a set of long test vectors to the device, instead a command, send via JTAG, starts the test. After test execution a single bit informs about pass or fail of the device. A standard BIST implementation does not have any diagnostics capability. Furthermore the time to test could be longer and the additional hardware overhead could be high for an exhaustive test. But such a test does not send data to the output ports, so that the risk for a security leakage is minimal.

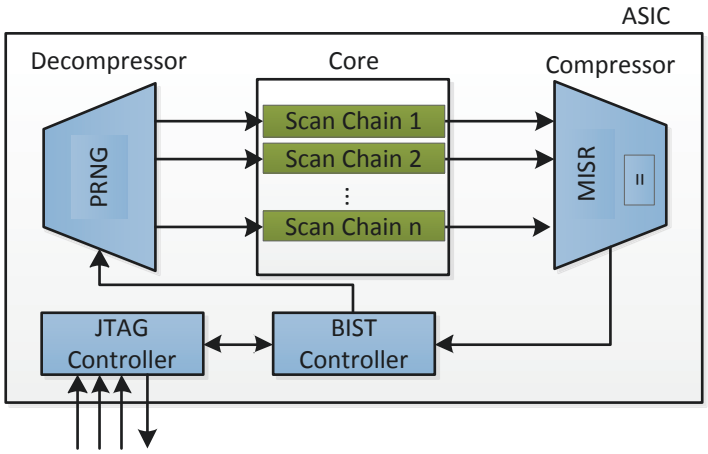


Figure 23: Core with all components, which are required for an offline BIST test. The additional components are marked in blue. A PRNG works as pattern generator and an MISR is a compressor unit to verify results. The JTAG interface is responsible for communication with test equipment.

Nevertheless a BIST is a useful replenishment to a scan test. To get a certificate of FIPS140-3, level 3, [57] beside an extensive post-production test, a self-checking of the device is required.

3.2 Scan Chain Attacks

Different scan chain attacks are published, for example in [87] and [54]. A simple scan chain attack writes and reads registers through the scan chain shift register, followed by a data extraction and analysis.

It is obvious from the description of scan test in Section 3.1.2 that all registers can be filled with attacker determined data. If the `scan_enable` signal is released and one clock cycle is performed, exactly one internal logic operation was executed. After such a working phase the data of all registers can be read out. This could be one possible and very simple attack by misuse the scan interface.

The CMOS technology has the property, the clock frequency of an ASIC can be reduced from its target frequency into the range of kHz or less. As long as the power supply is connected, the ASIC is in its current state including the data in SRAM and registers. So a low-budget microcontroller can be used to generate the signals for `clock`, `reset`, `scan_in`, `scan_enable` and `scan_out` to initiate an attack, even at low speed.

In Figure 24 a possible setup is shown. The host PC can be used to prepare input data. The microcontroller receives commands and data from the host, converts them into the protocol format of the scan chain interface. Finally the microcontroller forwards the output from the DUA to the host PC. There a comprehensive analysis of the data is possible.

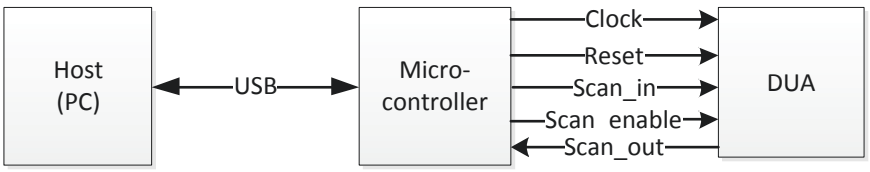


Figure 24: Simple setup for a scan chain attack with Host PC as user interface and microcontroller for signal generation.

3.2.1 Scan Chain Attack – Preparation and Duration

Since the debug interface of a device is often well documented, details of the scan chain are not typically published. Nevertheless, with additional information, which are public available, the structure of the scan chain can be determined. This is necessary to perform an attack.

The data sheet of an ASIC provides pin information, so that `scan_enable` pin can be identified with a high probability. Setting the `scan_enable` pin and a `clock` signal on the corresponding clock input, the attacker is able to find out the scan data out pins. On these pins a more or less random data pattern is visible. Now, the scan input pins are easy to identify. On each input pin of the device a predefined pattern, e.g. 0, 1, 0..., is applied. After a certain time, this pattern is shifted out on the scan data output pins. Now, the scan chain interface is fully determined. At this point the structure of the scan chain is still unknown. The length of each scan chain can be determined by applying a known pattern at the scan input and counting clock cycles until the pattern appears on the scan data output. In the last step a detailed analysis is necessary. Flip flops which are in focus of the attacker have to be identified. E.g. an encryption can be executed and the content of the flip flops is observed via scan chain to identify the flip flops which are involved into the encryption scheme. Based on this knowledge an attack can be performed, for example to extract the secret key.

To conclude the following steps are necessary to determine the scan chain properties:

1. Extract information from data sheet to identify `scan_enable` pin
2. Apply unique input pattern at each input and observe output pins
3. Assign scan input to corresponding scan output
4. Determine length of each scan chain
5. Assign relevant flip flops to its property and position in scan chain

3.2.2 Example of Scan Chain Attack

It is important to know, how a scan chain attack can be performed to get a feeling, why an unprotected scan chain is dangerous. In [87] the authors show a straight-forward attack on a DES implementation. But the DES algorithm is outdated, so in the following an attack onto a secure cipher algorithm is described.

Starting point for an example scan chain attack is a straight forward AES implementation with a scan chain. Clearly all other types of cryptographic algorithms are vulnerable for such an

attack if no countermeasures are applied. The AES algorithm is explained in [40] and the chosen implementation for the attack is described in [86]. The AES in this example is implemented for WSN and the secret key is stored in a NVM and the NVM cannot be read from outside. The area is very small and every step of the algorithm is performed in one clock cycle as shown in Figure 25. In the first clock cycle the AddKey operation is done, in the next one the S-Box operation and so on.

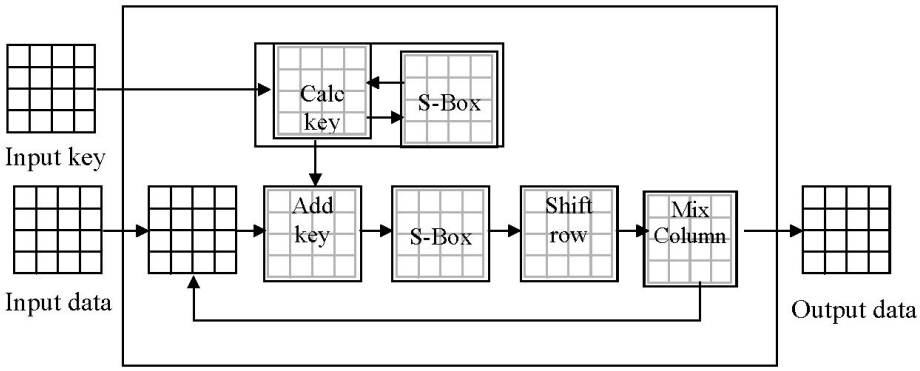


Figure 25: AES Implementation. Data as well as key are processed in a 4x4 byte matrix. Starting point for attack is AddKey operation. Each arrow accords to one clock cycle.

A scan chain is inserted and it is possible to switch from functional mode to scan test without any restriction, which means that no countermeasures for scan chain attacks are integrated. From the view of an attacker this has the advantage that the encryption or decryption operation is interruptible after the first AddKey operation. The data register contains input data XOR the secret key. Since the input data is known, the key can be recalculated by an XOR operation with known input data. In addition the attack is supported by a poor implementation. A well designed implementation avoids such beginner's mistake.

To illustrate the simplicity of an attack on an unprotected device, a possible attack is described in detail: After a reset the data for encryption is written into the device. The best choice for the input data is "0...0". The structure of the scan chain is unknown at the moment, so the data is written in using the standard interface. After one clock cycle the `clock` is stopped, because in the current state the flip flops contain data XOR key. With use of the scan chain the register values are shifted out. This intermediate value is the secret key, but the bit order is not known at the moment. To determine the correct bit order, in a next step a chosen plain text attack is required. The sequence is the same as in the first step. A data block for encryption is written into the core, after one clock cycle, the intermediate result is read out per scan interface. In sum 128 test patterns are necessary and in each pattern one bit position is set on "1" and all others on "0". A short analysis identifies the corresponding bit position of the key in the output data.

This straight forward attack is quite simple. With well-prepared equipment and an existing setup, such an attack can be performed in some seconds for a known device. The attack can be done by using a cheap microcontroller, which controls the scan interface. High-end equipment

like a high resolution oscilloscope is not required, because all signals are digital and no analogue data has to be analyzed.

3.2.3 Complex Scan Chain Attacks

On first inspection a scan test approach with test response compaction is not susceptible to scan chain attacks. However, in [17] a way is shown to get the secret key from a device, which uses test response compaction. The effort for such an attack is much higher than described in Section 3.2 and consists of four steps:

1. Create device model for logic simulation
2. Create look-up table with secret key values and corresponding compacted test response
3. Perform scan test and read out compacted test response
4. Search in the look-up table to get the secret key

The disadvantages, from attackers view, are the complexity and that the implementation details and the used test compaction scheme have to be known. In case of modern cryptographic algorithms with a long key, like AES or ECC, the look-up table would be so large that is impossible to create it. Nevertheless, such an attack is possible, e.g. for cryptographic algorithms with short keys or in case the key handling is divided in several parts.

3.2.4 Board Internal Scan Chain Attack

In case of scan chain attacks mostly external attacks are in mind. This means that an attacker uses, e.g. the JTAG connector on PCB to manipulate the ASIC or to get secret information. A complex PCB can have several ASICs which are combined via JTAG to fulfill test requirements. It is possible that one or two of these ASICs are owned by an attacker as shown on Figure 26. So an attack on an ASIC on a PCB is possible without any external access. It causes that the following attacks are possible, which are described in [63] in detail:

- Sniff and store signals on JTAG path
- Modify signals on JTAG path during test
- Initiate communication between devices

To sniff the data from the JTAG chain is a very simple task, if the attacker's ASIC is part of the JTAG chain. During a scan test several megabytes of data are transferred. Such an amount of data can be stored in a NVM for later processing and analysis, because the memories which are on the market are very large. During post-production test the ASIC of the attacker is able to modify the input data and the response values of all other ASICs on the PCB. So, e.g., an error affected device can be declared as error-free. The most dangerous situation comes up, if one device initiates a communication using the JTAG interface while the application runs. A secret key can be in a register which is accessible via JTAG and so the attacker is able to get the secret key.

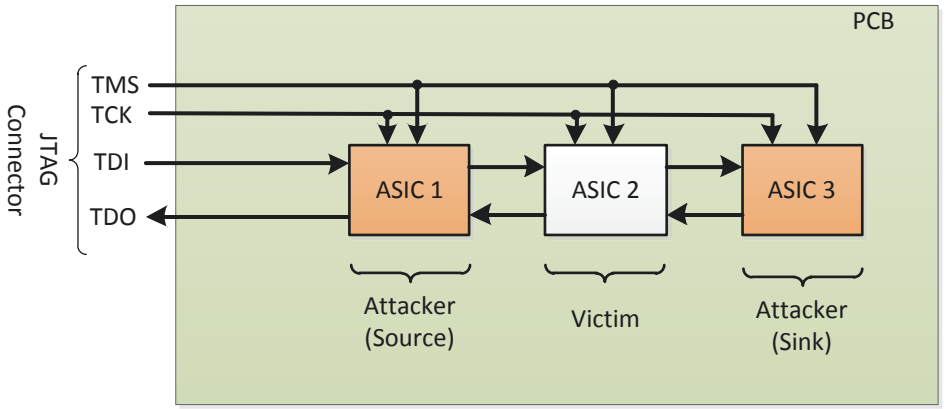


Figure 26: JTAG PCB internal attack. The first involved attacker sends stimulating data to the victim and the second involved attacker stores the output of the victim in its memory. Figure adapted from [63].

3.2.5 Conclusion

Scan chain attacks against unprotected devices are easy to perform. A digital interface is used to stimulate the design as well as to read out data. In most other Side-Channel Analysis (SCA) like Electro Magnetic (EM) attack or measuring the power consumption, the output from the measurement is an analogue value. It has to be interpreted and mapped to digital signals. The probability of an error in this step is high because the interpretation is based on a noisy analogue value. In case of a direct read access of the scan chain interface, the output is digital already and it cannot be interpreted in a wrong manner. Even unusual attacks are possible like the board internal attack, in which ASICs as part of the JTAG chain are involved.

3.3 Secure Scan Techniques, Countermeasures and Attacks

Countermeasures against scan chain attacks are manifold. They are divided into destructive and non-destructive protection mechanisms and Figure 27 gives an overview. Every solution has its own advantages and disadvantages. A destroying approach like disable the scan chain interface by a fuse bit is highly secure. A type of physical attack is necessary to re-enable the scan chain interface.

Other solutions require a secret key to enable the interface or the output of the scan chain is scrambled in a pseudo-random way. The most complex solutions are using an authentication scheme with well-defined standardized cryptographic algorithms. All these solutions have in common that additional hardware is required to establish a protection mechanism against side channel attacks via the scan chain interface. A general overview of existing solutions is given in [29], [11] and [16].

Possible countermeasures, which are explained and compared in detail below, are:

- Logical modified scan chain

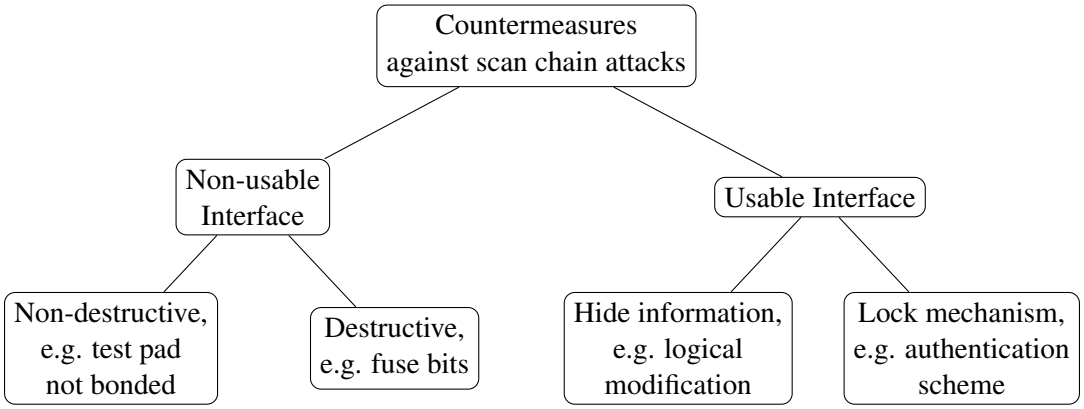


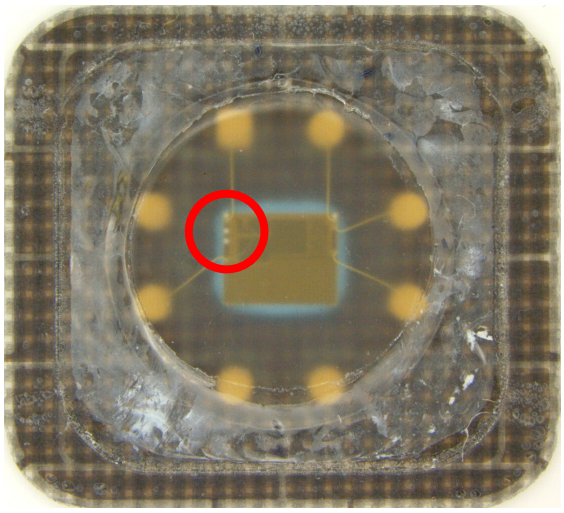
Figure 27: Classification of countermeasures against scan chain attacks.

- Fuse bit
- Non-accessible key
- Compression technique
- Lock and key technique
- Authentication based security mechanism
- Encrypted scan chain channel
- BIST

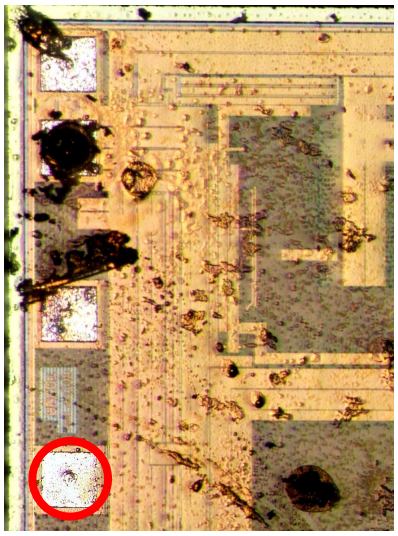
3.3.1 Preventing Physical Access

The first idea to prevent a scan chain of undesired access is to prevent the access to the test pads. The test of the DUT is done at wafer level and in the following packaging step the scan chain pads are not connected to an external pin by a bond wire. Unfortunately this countermeasure can be bypassed very easy. An example for this attack is given in Figure 28 and this device is extracted from a smart card. The function of every external pin is documented in a standard [34]. At first the cover, made of plastic is removed and the die can be seen as shown in Figure 28a. An inspection under a microscope shows unbounded pads, which are probably pins for testing. The device can be unpacked, as shown in Figure 28 and all pads are bonded on a small PCB with low effort. After rebonding the test pins are accessible. The pin assignment of the non-documented pins can be identified by an experienced engineer, as described in Section 3.2 and internal information can be read out.

A further option after a wafer level test is cutting the test pads. This solution is not suitable for many designs. A whole edge has to be cut, which can be a lot of silicon area and this is expensive. In Figure 29 the design principle is shown. The I/O pad ring is extended by an additional row with three pads which are required for a scan chain test. After the test these pads are cut. In the given example in Figure 29 the die has a size of $4\,\mathrm{mm}^2$ and additional $0.3\,\mathrm{mm}^2$ are required for the test pads.



(a) ASIC of a smart card still covered with epoxy. Some pins are bonded with wires and some not bonded test pins, which are marked in red.



(b) Fully unpackaged ASIC of a smart card. The epoxy and bond wires are removed. The small dot, marked in red, at the lower pad comes from a needle, which was used in post-production test.

Figure 28: Unpackaged smart card

3.3.2 Scan Detection

While all other solutions proposed are countermeasures against scan chain attacks, the spy flip flop [28] is a detection circuit only. The designer has to decide, which action is done in case of an unexpected scan test, which could be a scan chain attack. Possible activities inside the ASIC are deletion of secret key memory or a general reset of the device.

Figure 30 shows a simple but effective solution to detect events along the scan chain. The shift detection circuit consists of a flip flop with a fixed input value (e.g. “1”), which is part of the scan chain. The output of the flip flop is input to combinatorial logic. This logic tests the output on “1” or “0”. The value “0” can only be reached if a flip flop of the scan shift register has a “0” propagated to the spy flip flop. To ensure that a scan chain attack can be recognized for both cases (“0” and “1”) a detection circuit should be integrated. The integration is possible at RTL so it is technology independent with a very low area overhead. Nevertheless it is not clear why the spy flip flops are necessary because the scan chain itself is enabled by the scan enable signal only. So it is sufficient to observe the scan enable signal.

3.3.3 Modified scan chain

This security mechanism is based on “security by obscurity”. The input and output pins of the scan chain are not disabled. In fact the input and output values are different from a design

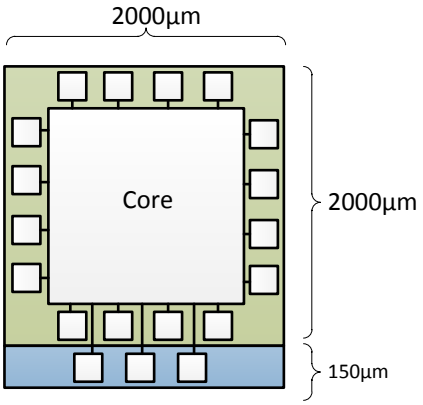


Figure 29: Principle of test pads to cut. The pad ring in green is the I/O interface of the ASIC and the additional row (marked in blue) contains the test pads. They are cut after the test.

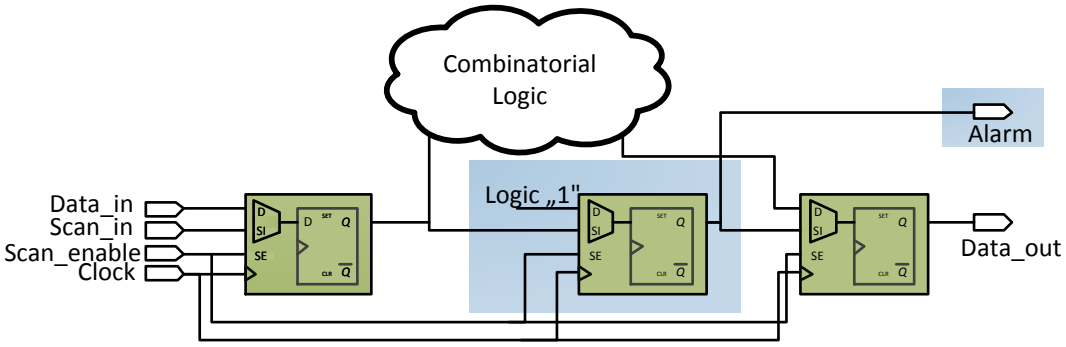


Figure 30: Detection circuit for scan chain use. The additional scan flip flop with a fixed data input, marked in blue, sets an alert if the shift register mode is used. Figure adapted from [28].

without the logical modified scan chain. Since the functionality of the ASIC is not changed, the long shift register of the scan chain is adapted. An attacker is confused because a mapping between input and output data does not deliver meaningful results.

A solution for a secure scan test with a very low area overhead is proposed in [66]. The scan chain is extended by inverters in between some flip flops to influence the input and output data of the scan chain as shown in Figure 31. The designer knows the positions of them and is able to adapt the test patterns. The inverter in front of the flip flop requires that the corresponding bit value, and all following, in the test pattern are flipped. The functional operation mode is not influenced in this case with one exception. Some designs use shift registers, e.g. in an SPI [52]. Scan insertion tools are optimized so that for a shift register no special scan chain is

inserted. Instead the shift register itself is used. The DFT designer has to consider this point and to exclude the addition of inverters in a functional shift register.

For an attacker it is possible to identify the inverters as part of the scan chain in the following way. At first a chain of zeros is written into the core in scan mode. In the next step the functional mode is used. A simple read operation on readable registers delivers a zero for registers without inverter in front of the flip flop and one for registers with an inverter in front of the flip flop. Furthermore the approach works as "security by obscurity" so that in case of reverse engineering the inverters are visible and it can be considered by the attacker. During the implementation the designer has to keep in mind that the number of inserted inverters is even. Otherwise if an odd number of inverters is used, is very obvious for an attacker that there is a countermeasure built-in.

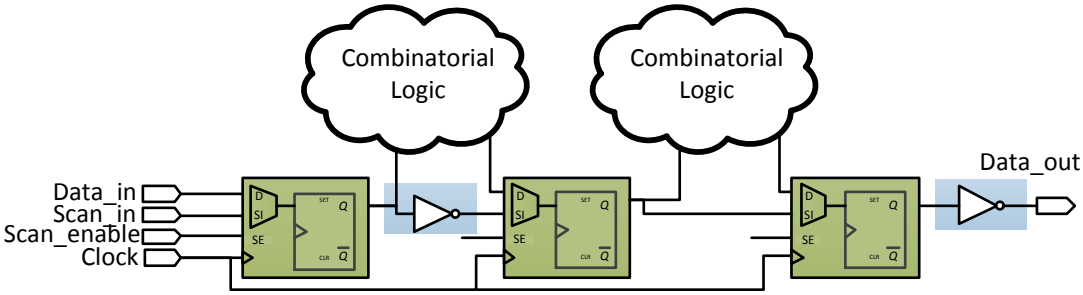


Figure 31: Logical obfuscation of the scan chain. Inverters, marked in blue, are inserted into the shift register path. An attacker does not know the number and the position of the inverters and the output of the scan chain is useless, because the data is modified. Figure adapted from [66].

3.3.4 Fuse Bit

A fuse bit is a suitable solution to deactivate a scan chain after the test procedure. It does not matter, whether a fuse or an anti-fuse is used for this purpose. Both types of fuses is common that they are blown once, the process cannot be reversed. In [58] the authors show a security mechanism, in which the `scan_enable` signal and/or the scan input and output are protected with the use of a fuse element. In Figure 32 an example is shown, in which every `scan_enable` input is protected by a fuse element.

In [53] the use of one or more EEPROM cells is described. Unfortunately the stored bits in an EEPROM can be erased, even if countermeasures are implemented. X-ray radiation can be used to delete the content of EEPROM cells [38]. The trapped electrons in the floating-gate transistor are removed during an x-ray treatment.

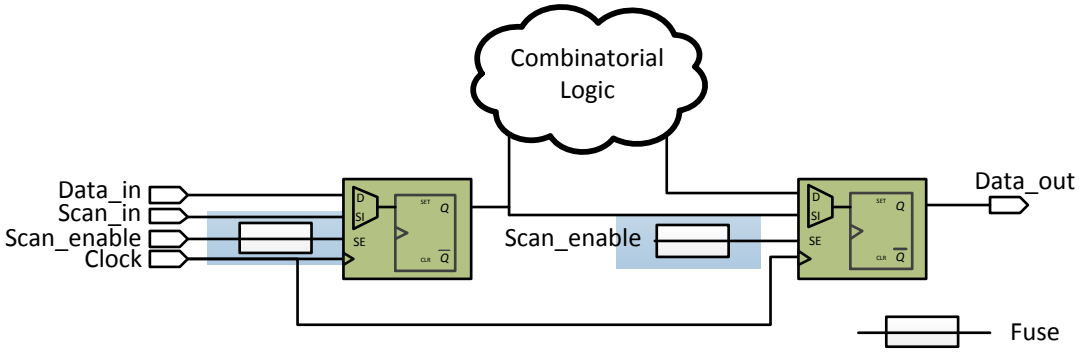


Figure 32: Fuse cells, marked in blue, for `scan_enable` input for each scan flip flop to deactivate the scan functionality after post production test. Figure reprinted from [58].

3.3.5 Non-accessible Secret Information

This type of countermeasure prevents a direct access to the secret key register of a cryptographic core, because they are not part of the scan chain. In the following two different approaches are described.

The Mirror Key Register (MKR) approach integrates a mirrored register as shown in Figure 33, which replaces the secret key register during a scan test. The proposed solution in [88] and [89] is declared as solution for cryptographic cores. If a scan test is performed the key for the test comes from the MKR. The secret key from system mode is never used in test mode. Therefore it is protected against scan test attacks.

This solution is implementable on RTL and it does not require any modification of the synthesized netlist. The MKR approach is well realizable, because the control structure is straightforward and the complexity is low. The scan patterns can be generated by standard software tools. Furthermore the power consumption is not influenced significantly, if clock gating is implemented. In a reverse engineering step the MKR is visible. But for an attacker this information is not helpful to disable the security feature. There is no secret key used and reverse engineering does not help to bypass this countermeasure. However, there is one exception for the security of reverse engineering. In case that the secret key for system mode is stored in combinational logic, it can be read out. But this problem occurs in all situations if the secret key is stored in that way.

The disadvantage of the solution is that the application area is restricted to cryptographic cores, especially to standalone modules. For SoCs, e.g., a processor and connected cryptographic cores, this approach is not suitable, since an interaction between the processor and the cryptographic core is possible so that a copy of the secret key is stored in registers of the processor.

Moreover, the use of MKR decreases the test coverage. In a typical scan test, the flip flops are tested implicitly. In this solution, only the mirror key registers are tested but not the register with the secret key. To test these register, a functional test has to be performed additionally.

Depending on the implementation the additional area is up to several per cent. For example the size of the AES implementation, described in [86] would be increased by $8\,\%$. The size of the core is $323{,}000\,\mu\mathrm{m}^2$ in IHP $0.25\,\mu\mathrm{m}$ technology. For an MKR, one multiplexer and one register are required for each bit of the 128 bit secret key. A flip flop has as an area of $155\,\mu\mathrm{m}^2$ and a multiplexer has an area of $49\,\mu\mathrm{m}^2$. In sum the additional area is more than $26{,}000\,\mu\mathrm{m}^2$. Furthermore, a speed test is not possible, since of the logic paths in test mode are not equal to the logic paths in non-test mode.

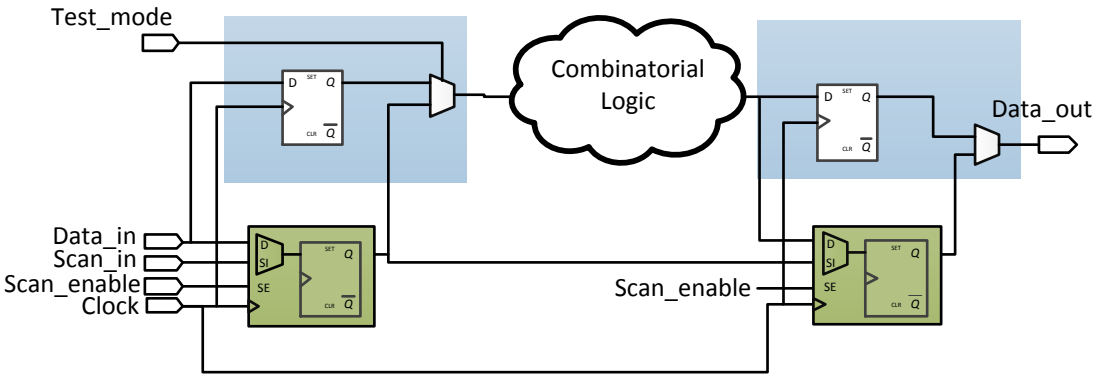


Figure 33: The design contains two types of flip flops. The scan flip flops are used during test mode, the flip flops without scan functionality, marked in blue, are used in operation mode. Figure adapted from [88].

In contrast to MKR the solution in [71], shown in Figure 34, does not use a register to protect the AES core. In case that the JTAG interface gets a scan test request, a fake key is loaded into the secret key register. Due to the fact, that the fake key is a fixed value with zeros for all key bits, the area overhead is very small. Furthermore the authors implemented a special function, which prevents interrupting of an ongoing encryption. So, an access to intermediate results is not possible, which would be a leakage. The authors describe, that the use of a fixed fake key has the disadvantage that the key is fixed during all tests. So they evaluated different keys regarding the test coverage and in the end, a fixed key with all zeros results in the highest test coverage. Independent of the small area overhead this solution seems not very practicable because the test coverage is decreased slightly. The process for the search of a key with high test coverage is a very time consuming. Furthermore the solution is not generic, so that is usable for cryptographic cores.

3.3.6 Compression Techniques

The characteristic of compression technique is that scan data input and output are compressed. In contrast to the BIST the input and output data are feed into the DUT and they are not generated inside of the ASIC. A compression for output data reduces the amount of direct analyzable information. The highest compression rate would be achieved by a single bit output for the

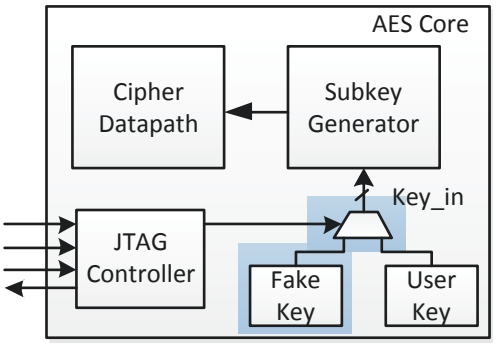


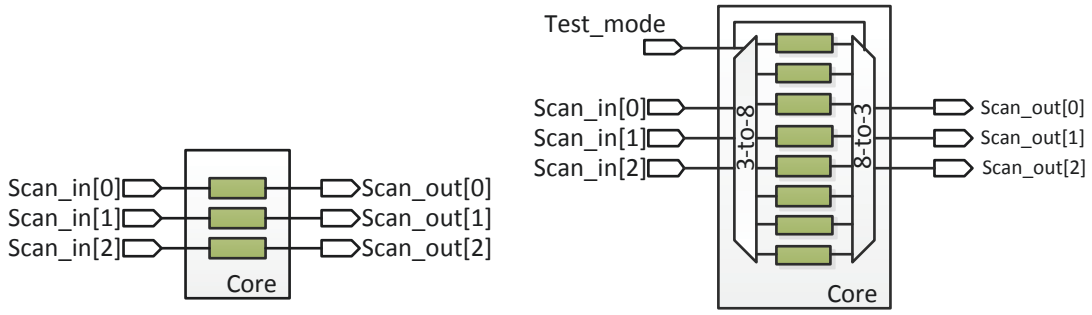
Figure 34: The AES implementation is enhanced by an additional multiplexer and the fake key component. In case of a scan chain test, the JTAG controller selects the fake key as input for the subkey generator. The additional components for the secure scan test are marked in blue.

whole test. A “0” or “1” informs about an unsuccessful or successful test. The amount of additional information for the user as well as for the attacker is equal to zero, because this is a lossy compression technique. The security level is very high, but a fault analysis is not possible.

In case that the compression algorithm is known and the compression is lossless, it is possible to gain information of the uncompressed data and to perform known scan chain attacks. In Figure 35 a very simple compression mechanism is shown. This approach does not provide any security. A design with a given number of scan chains is modified to a design, which contains more of them. The number of scan in and out pins is equal. A compression and decompression unit is added to the design. E.g., in the example given in Figure 35, a 3 bit input drives 8 scan chains. The compression causes a reduction of the required scan vectors. The components are inserted by the DFT tool and the compression scheme is well documented. A successful scan chain attack against an AES implementation with a compression scheme is described in [22].

A compression test for an AES in a pipelined version is shown in [67]. The implementation consists of ten identical modules, one for each round. Due to the fact that the scan chain in every module is the same, the scan data input and scan data output is the same, too. A comparator checks the output of every module as shown in Figure 36. In case of any deviation an error the compare unit sets a corresponding signal. The error signal allows identifying the test vector, but not the module which fails. A diagnostic capability can be added by an additional component. This component allows enabling of the scan test for each module separately. In case of an error, each module can be excluded from the scan test. If the defect module is disabled, no error is signalized.

The advantage of this solution is that very few information is available for an attacker and direct read access to intermediate results is not possible. A special design helps to identify which round module is responsible for the error and which logic gate inside of the round module has been damaged during production. Since every round module is equal to the other modules,



(a) Multi scan chain approach without compression.

(b) Scan compression, in which 3 inputs and outputs are mapped to 8 internal scan chains. A faster and efficient scan test is possible, without any protection mechanism.

Figure 35: Comparison: scan chain vs. compressed scan chain.

the time for generating test patterns is decreased significantly, because the test pattern are the same for all modules. Even the time to test is reduced.

The disadvantage is that the solution is only suitable for circuits having a regular structure with equal modules inside. Every module has to be fully identical to the other ones. Not only the logic structure, but also connection between the scan flip flops. For designs, which are not timing critical, the schematic of each module is equal to the other one. But in case of a timing critical design, a reordering of the scan chain is a possible optimization. Thereby in layout process the serial scan path between all flip flops is adapted. As result the schematic is changed slightly. Nevertheless such a small modification prohibits the use of the described compression method.

3.3.7 Lock and Key Technique

The main characteristic of the lock and key technique is the use of a secret key. Every countermeasure, which enables access to the test port with a password, is classified as “lock and key technique”.

Two different approaches are known. In the first case the device contains a golden key and only if the test engineer enters the correct key, the interface is enabled. In the second case the interface is always on. The secret key is used to encrypt and decrypt the input and output data of the scan interface. A similar technique uses public private key authentication. Such techniques are described in Section 3.3.8 in detail.

In [41] and [42] the author describes a key-based authentication scheme for the scan test. Without the correct secret key, the output of the scan chains is scrambled. The scan chain is divided into several sub chains. Although the design has internally many sub chains the interface to the outside is the same as before as shown in Figure 37. The additional scrambler unit is responsible for sorting the output of the sub chains. To test the device the secret key is

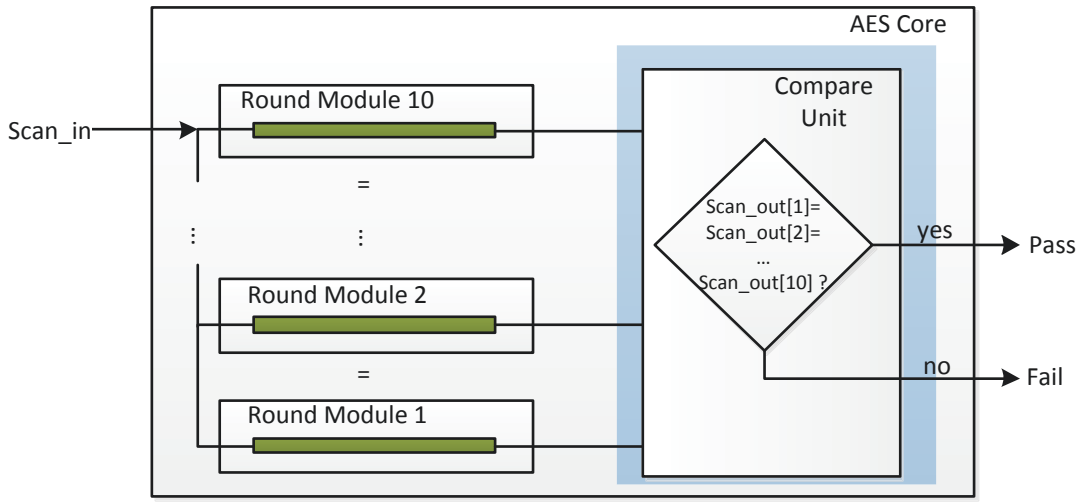


Figure 36: Pipelined AES implementation, one module for each round, and an internal compare unit, marked in blue, for the scan chain output. The compare unit compresses the output to a minimum value. Only a pass or fail signal is generated. Figure reprinted from [67].

put into the device. If the secret key is correct, the serial data stream at the scan data pin is right order. In the case that the secret key is wrong the output of all sub chains is scrambled by the multiplexer, which is controlled by an LFSR. In the case of multiple scan chains, this step is performed for each scan chain separately.

The protection mechanism has two different states. On the one hand the insecure state, this is the default after a reset of the device. On the other hand the secure state which is entered after the correct secret key is put into the device. The LFSR is the key feature of the proposed solution and it controls a multiplexer. The multiplexer combines all sub chains to a single propagated output. One requirement of the LFSR is that every sub chain has to be selected and its output is propagated to the scan data out before any sub chain is selected again. Otherwise the time to test increases significantly.

The advantage of this solution is that test pattern generation is possible using standard tools and a post processing is not required. Only one additional test vector has to be generated which writes the secret key before the scan test is initiated. If the correct key is written into the device the output vectors on the scan data output pins are equal to the generated vectors from the test pattern generator. The additional overhead is restricted to the additional time which is required at the beginning of the scan test to feed the secret key into the device. The application of this solution requires an adaption of the netlist. One scan chain is replaced by several scan chains and an LFSR and a controller is added. Changing the netlist can cause some errors, so comprehensive verification is necessary. In case of a wrong key the input and output is not blocked and data can

be shifted in and out. In principle due to the scrambled scan chain the output data are useless, but in best case no data is available at the scan data output pins.

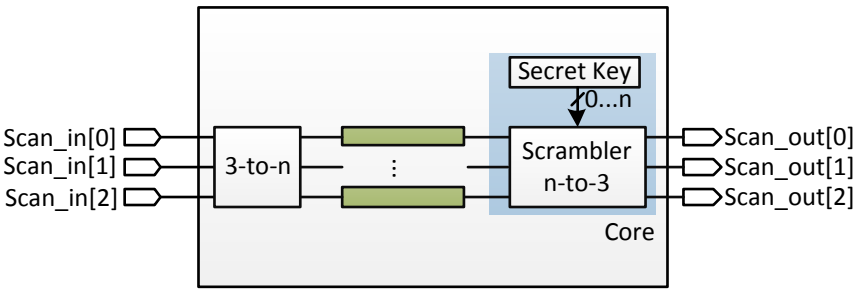


Figure 37: Lock and key technique for scan chain protection. Without the correct secret key, the output of all scan chains is scrambled in a pseudo-random way. Figure adapted from [41].

The proposed solution has disadvantages. At first the system is not safe against reverse engineering. The implementation is time-consuming because a suitable LFSR has to be found. Furthermore a wrong secret key does not lock the input as well as the output of the test interface. Therefore it is possible to write and read data although the data is scrambled. This data can be used to perform an attack.

In [11] a solution for the JTAG standard IEEE 1500 is shown. In Figure 38 an overview of the solution is given. A direct access from the boundary scan register to the scan chain is removed. At the beginning of the scan chain test using the JTAG interface, a secret key has to be written into the device. This key is compared to the internal golden key. In case of a granted access the AND gates get a logical “1” from the security controller and the access to the internal signals is possible. To keep the additional overhead as low as possible, the golden key is not stored directly within the device. Instead the golden key is generated using an LFSR in combination with the registers which are already part of the JTAG controller. The advantage of this solution, is the sharing of existing resources (flip flops). Therefore the overhead is very low. Diagnostic tests are possible to identify production defects. The disadvantage is that the interface is usable at any time if the secret key is known. Furthermore a reverse engineering is possible. In this step the implementation of the LFSR can be extracted and the secret key can be determined.

3.3.8 Authentication based Security Mechanism

In an authentication based security mechanism, the device contains a public key and the test engineer has the corresponding private key. Only with a matching key pair the access is granted. Compared to the lock and key technique from Section 3.3.7 the solution is robust against reverse engineering. An extracted public key is useless, since the corresponding private key cannot be derived from the public key.

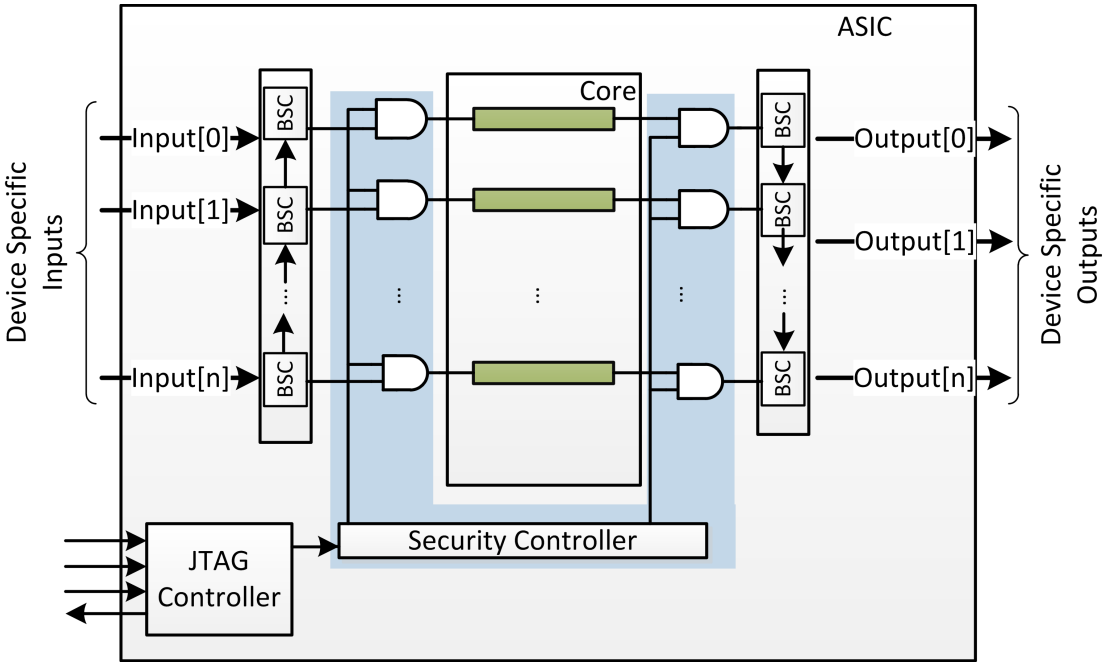


Figure 38: Lock and key technique for IEEE 1500. Additional AND gates between boundary scan register and core are used to control the access to the core internals. The additional required components to implement the security mechanism are marked in blue. Figure adapted from [11].

In [60] and [61] an authentication based security system is introduced, which uses the JTAG interface. The approach is shown in Figure 39. The access monitors allow fine granular right management so that e.g. one user is allowed to use the scan chain but another user is only able to update the content of a NVM. The approach can be used with different authentication schemes and it is proposed to use a server based authentication protocol, which is shown in Figure 40. Such an adaptable dynamic user management allows to drop out a doubtful user from the list, restrict the access or to enhance the rights of a user. It is required that a private key is stored into the device. A controller is responsible for the protocol handling. Therefore the individual right management provides as many privilege levels as required. Each scan chain has its own security mechanism and a memory controller restricts the access according to the privilege level for each address range. An access monitor enables the corresponding components. The required area is significant. For this specific implementation an area of 74,681 GE is used.

In [19] a Physical Unclonable Function (PUF)-based authentication is presented. Beside the PUF generator, a PRNG, memory and controller are required. After production every device generates a challenge-response pair and the challenge is a random value. The response of the device is based on the challenge and PUF value. After this step the read access to the PUF element is disabled, e.g., by blowing a fuse bit. The authentication protocol was developed in such a way, that it prevents replay attacks. Since replay attacks are not possible, this solution is suitable to avoid attacks on a PCB as described in Section 3.2.4. The authors of [19] describe the

solution explicitly for cryptographic cores, but the approach is applicable to any design. The area overhead is small, around 3,000 GE, and the timing of the scan chain itself is not influenced. Although the approach is for an SoC, the enroll phase seems to be very expensive and time-consuming. Furthermore, a PUF is required and the test engineer has to trust the PUF database. However, it is a secure solution also for various devices in one JTAG onto a PCB.

A further authentication mechanism is described in [18]. An ECC based protocol is implemented and allows an authentication of the device as well as the user. The hardware implementation of the ECC core has a very high area overhead and the initial setup time is high too. After finishing the complex protocol the input and output of the scan chain is handled without any encryption.

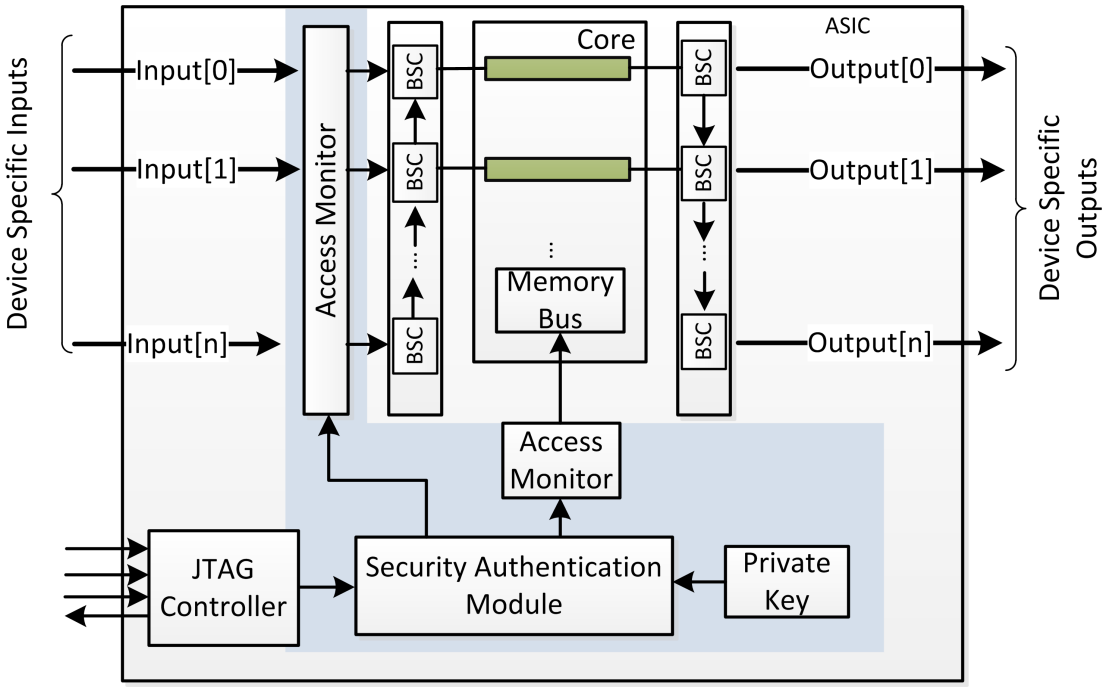


Figure 39: Authentication based security extension for a JTAG interface. The additional components are marked in blue and they offer a fine granular access management due to the use of an integrated access monitor for the memory bus. Figure adapted from [60].

3.3.9 Encrypted Scan Channel

As described in Section 3.2 attacks against a PCB along a JTAG chain are possible. To get a secure test environment for specific devices on the PCB a high effort is required [63]. A full encrypted channel is established between the DUT and the test environment. This includes:

- Authentication
- Encrypted communication channel
- Integrity check

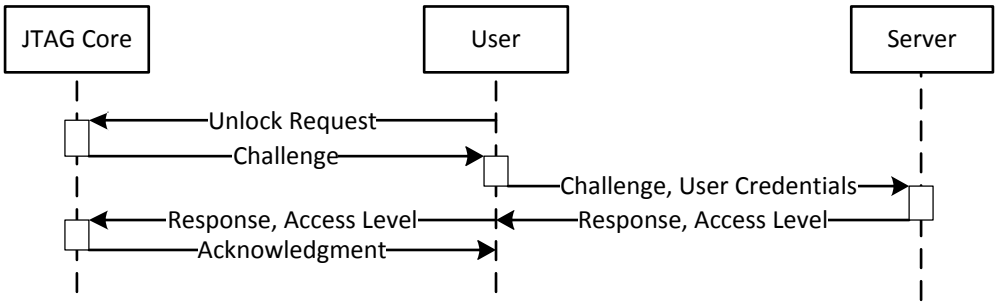


Figure 40: Message exchange for authentication based security extension for a JTAG interface to establish an access to internal scan chain and memory. Figure reprinted from [60].

Fuse bits, which are set in the factory, generate a unique identifier. A feature is that each device in the JTAG chain establishes a fully encrypted data channel to the test environment. In the setup phase the secret keys between test environment and ASIC are shared. In the communication phase the data stream between tester and ASIC is encrypted and a Message Authentication Code (MAC) is added to verify the source of the data. Different types of attacks like sniffing of data or man-in-the-middle are effectively prevented using this countermeasure.

The additional area overhead is high, more than 126,000 GE are required. Furthermore an initial communication protocol is needed to establish the secured channel before every test. For this, a time of roughly 2,200 clock cycles has to be considered.

3.3.10 Offline BIST

A BIST is a well-known test technique as described in Section 3.1.3. The communication to the test environment is restricted to few signals like start, stop and fail.

An example for an offline BIST for a general purpose CPU is given in [74]. The authors use modules which are available in the device to generate test pattern and to compact the test result. The generated test patterns are a serial bit stream which is used as stimuli of an internal scan chain. The idea is usable in a design with a general purpose CPU, because an accumulator and a register with feedback are required. Even the response compaction after the internal scan test is done using an adder and register of the CPU. This means that the adder-register unit or the register has to be twice in a circuit to generate the test pattern and to perform the response compaction. The disadvantages are that the adder and the register cannot be part of the scan chain and the defect analysis is very complicated. Like in every other BIST the test pattern are generated internally. This causes that much test patterns have to be generated to reach high test coverage.

In [65] a BIST is analyzed with focus on cryptographic cores in general. The block cipher 3WAY [15] is used to show an example implementation for this test approach. An LFSR as shown in Figure 41 is used to generate the pseudo random input data and a control unit feeds this data into the key generator and into the data registers. An encryption or decryption is performed and the result is fed into a MISR and the result is compressed and shifted out. This output

is compared to expected data and the ASIC can be classified as “pass” or “fail”. The authors describe that 20 encryption and 20 decryption operations are required to reach a test coverage of more than 99 %. The advantage of the solution is, that no data transfer in type of a scan chain is necessary. In contrast the disadvantage is that the approach is specialized for cryptographic cores and not for general purpose CPUs. Furthermore the design has to be analyzed with a high effort to get a high test coverage and not in every case such a high test coverage seems to be possible.

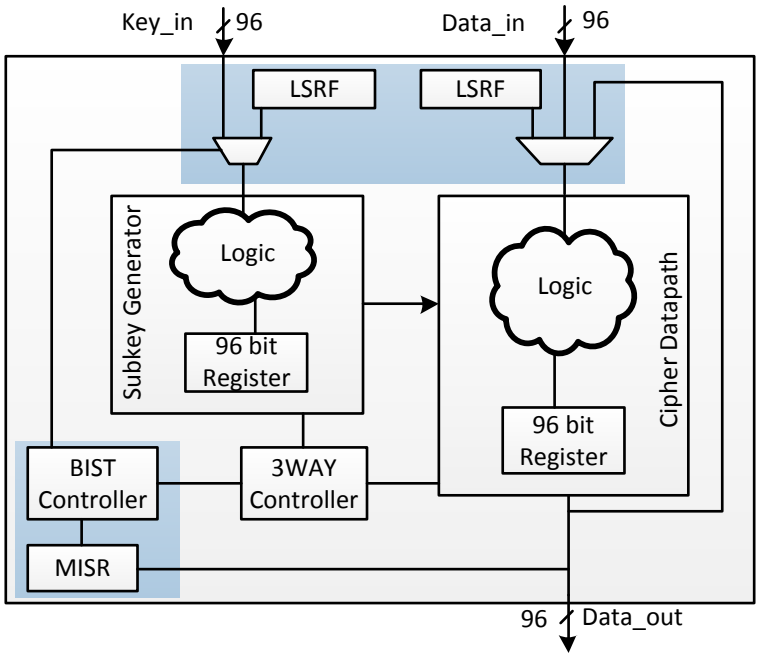


Figure 41: Offline BIST integrated in 3WAY block cipher. Additional components for test are marked in blue. LFSRs generate input data and MISR compresses the data output. Figure adapted from [65].

3.3.11 Online BIST

IP core dependent solutions are introduced here which are not based on scan chain test. For example in [56] an on-line self-test of an AES core is described. The assumption is that the S-Boxes are the main element within the design and have to be checked for errors. The authors chose an AES implementation in which the S-Box and the flip flops are combined in one module and them argument that this module requires 85 % of the area. The later parts of the AES algorithm are AddKey, ShiftRow and MixColumn and they are untested.

Four S-Boxes are implemented in the design. For the on-line test, an additional S-Box with flip flops is added as shown in Figure 42. This S-Box is used to check the other ones. It gets

the same input data as the S-Box which should be tested. Both outputs are compared. In case of different results an error occurred. To test every S-Box the additional S-Box checks all S-Boxes in the design in a cyclic way.

The advantage of this solution is a fast on-line test, but the disadvantages are dominant. The test coverage is by far less than 99 %, an error analysis is not possible and the solution is specialized for the AES. The additional overhead is more than 21 % in contrast to an off-line scan test which requires an additional overhead of less than 10 %.

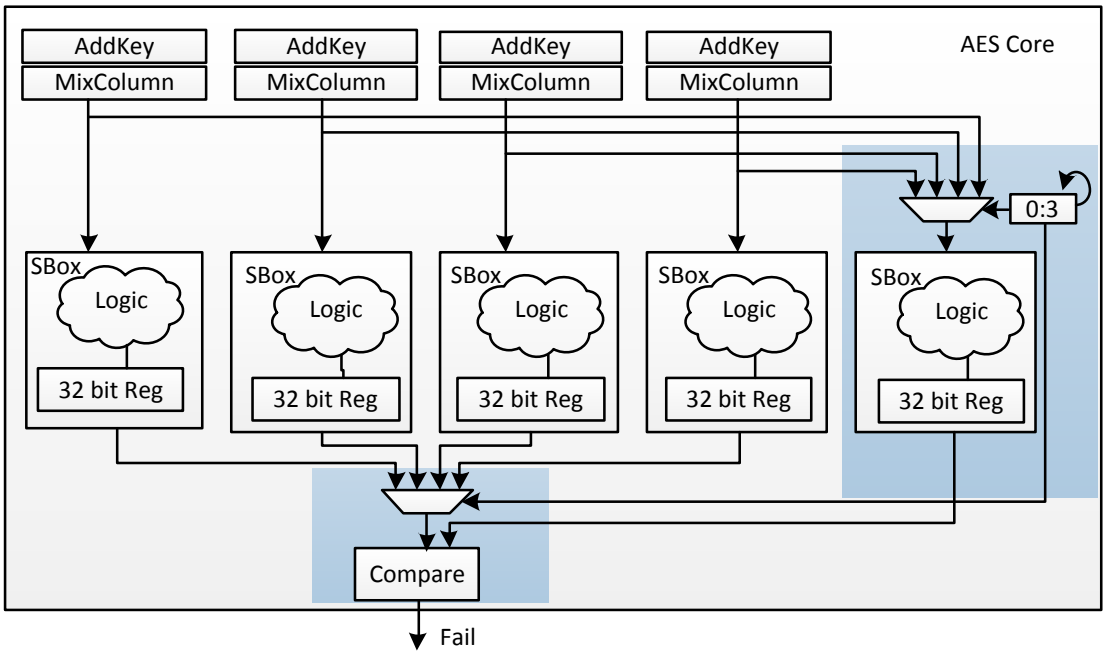


Figure 42: Online BIST in an AES implementation. A redundant S-Box and a compare unit, marked in blue, are added to the design. All four S-Boxes are tested in a cyclic way. Figure adapted from [56].

3.3.12 Discussion

In Table 1 countermeasures against scan chain attacks and their evaluation are shown. Different types of countermeasures are available. Each of the described approaches has its advantages as well as disadvantages.

A countermeasure which is independent of the design can be easy ported to another system. For example the authentication based scan chain interface with ECC is a stand-alone component with a single output to enable and disable the interface. It can be used to protect any system. In contrast to that generic protection mechanism, the online BIST [56] is very design specific. A change of the design requires a re-design of the BIST module. Even the implementation effort

differs with the type of protection mechanism. E.g. a fuse bit is very simple, but an authentication protocol requires a lot of time and experience to implement. The silicon area is an important fact. Each fabricated device contains the countermeasure. Especially in a cost-sensitive area for mass products the area overhead should be as low as possible. The same statement is valid for the additional time to test the device. In some cases a time consuming protocol has to be executed before the scan chain test can be started, e.g. the ECC-based authentication requires up to $133\,\mathrm{ms}$. The solution with the scrambled output has a very low overhead regarding the time to test. Only the secret key has to be written into the device. This additional time is negligible. All solutions for a secure test have minor influence on the timing of the design. It means that the additional components do not influence the internal signal paths of the DUT in most cases. An example for a changed timing is the mirror key register. It influences the signal path slightly by insertion of a multiplexer into the signal path.

The test patterns generation is part of the DFT flow and it is done by a test pattern generator. This generator cannot be influenced with respect to a secured scan chain interface. If setting a secret key before the scan test patterns are applied, an additional set of test patterns has to be generated externally. For example the lock and key technique with the scrambled output requires a small additional pattern to set the secret key. In contrast to this simple mechanism the effort for the encrypted scan channel is much higher. The scan patterns are generated for a system without an encryption scheme at the scan chain interface. So a tool is required which reads in the scan pattern, encrypts the data and write out the encrypted scan chain patterns. Even some protection mechanisms are not suitable to protect a microcontroller system. E.g. the online BIST [56] is a very specialized solution for an predefined encryption algorithm. Other solutions like the authentication based mechanisms protect the scan chain interface, independent of the internal structure.

Even this online BIST has an influence on the test coverage. Depending of the implementation the test coverage differs. The protection mechanisms for the interface instead do not have any influence on the test coverage. Test structures which uses scan chain pattern allows a fault analysis. In case of a fabrication error the output pattern is not equal to the reference output pattern. Depending of the position in the data stream the defect can be localized. With a BIST based fabrication test such an analysis is not possible. A device without any protection mechanism for the scan chain interface can be tested immediately after fabrication. It means that there is no additional effort after fabrication for every device. In case of the PUF based authentication mechanism, after fabrication the PUF has to be read-out from every device and stored in a database. To test a device an access to the database is necessary. An error in the mapping of the device and the entry in the database causes that the ASICs are not testable. Especially in the area of microcontroller the JTAG interface is a common approach as interface for testing and debugging so that it useful that the countermeasure can be integrated into an JTAG interface.

Important facts are the properties of the protection mechanism against scan chain attacks regarding the security. The resistance against reverse engineering and probing attacks has to be checked. A protection mechanism like the PUF based authentication is a very complex mechanism with a high effort regarding area and additional time to enable the test procedure. But an attack using a needle probe is relatively simple. It is sufficient to identify the `enable` signal inside of the ASIC and to “overwrite” the signal with an external voltage source. Such an attack

can be performed by attackers of class 2. A laboratory with FIB, needle probe and voltage source might be available in universities and so on. The authentication solution in [61] avoids a single point of attack. So an attack is much more complicated and much more expensive with respect to money and time. E.g. the compression technique [67] is resistant against most of attackers. The result of the test does not give information about detailed test results. Only a pass or fail signal is enabled. In a reverse engineering step the implementation of the test mechanism can be extracted, but this is not helpful to attack the device. Nevertheless the protection mechanism is not usable for a microcontroller system.

In conclusion none of the shown solutions seem to be well suitable as countermeasure against scan chain attacks, especially with respect to the application area of wireless sensor nodes.

Table 1: Countermeasures against scan chain attacks and their features and properties.

Countermeasure	*Design independent*	*Implementation effort*	*Area overhead*	*Time to test* [1]	*Influence on design timing*	*Impact on DFT flow and preprocessing scan input data*	*Usability to protect a system*[2]	*Influence on test coverage*	*Fault analysis possible?*	*Initial effort after fabrication*[3]	*Usability for JTAG interface?*	*Resistant against reverse engineering*[4]	*Resistant against needle/probing attack*[5]	*Secure against attacker level*
Scan Detection[6] [28]	+	++	++	o	++	++	yes	no	yes	++	no	n/a	no	n/a
Modified scan chain [66]	-	o	++	o	++	++	yes	no	yes	++	yes	–	yes	1
Fuse bit	++	+	++	o	++	+	yes	no	yes	+	yes	++	no	1
Mirror key register [88]	-	+	-	++	+	++	no	yes	yes	++	no	++	yes	2
Compression technique [67]	+	+	+	++	++	++	no	no	no	++	no	++	yes	3
Lock and key: interface off[11]	++	+	+	o	++	o	yes	no	yes	+	yes	–	no	1
Lock and key: scrambled output[41]	–	+	+	o	++	o	yes	no	yes	+	yes	–	yes	1
Authentication [61]	o	–	+	-	++	o	yes	no	yes	++	yes	–	yes	3
Authentication PUF[19]	o	–	–	-	++	o	yes	no	yes	–	yes	++	yes	1
Authentication ECC[18]	++	–	–	–	++	o	yes	no	yes	++	yes	–	no	1
Encrypted Scan Channel [63]	+	–	–	-	++	–	yes	no	yes	+	yes	++	yes	3
BIST, offline[7]	-	+	o	+	+	++	yes	yes	no	++	yes	++	yes	3
BIST, online[56]	–	-	-	++	+	n/a	no	yes	no	++	no	++	yes	3

Legend:
++ very good + good o average - bad – very bad

[1]Single scan chain is reference timing
[2]System: CPU, cryptographic modules and memory
[3]E.g. read out a PUF-based secret key and storage in a database
[4]Information benefit to get access to the secure scan chain interface
[5]Read or influence internal signal
[6]Detection circuit only
[7]Tool supported integration

3.4 Programming and Debug Interface

In the following the state of the art in programming and debug interface, and securing it, is described. This type of interface is a common approach to upload and update and debug the software of a microcontroller. An abandonment of such an interface is almost impossible.

There is a difference between debugging and programming, which impacts to the security mechanism for the debug interface. In case that a microcontroller should get an update of the application program, the device can be rebooted before the programming sequence is started. The application program can be interrupted because after a code update it has to be restarted anyway. In contrast to a reprogramming for debugging a reset of the device should be avoided in some cases. E.g. if the device is in an error state it is useful to enable a debug session without a reset. The reset causes that the program is interrupted and the register file gets an initial value. So the information is lost which are required to debug a running program. In many cases it is very difficult and time consuming for the software developer to reproduce the error state again.

Since the process of debugging contains a debug and a reprogramming step, in the following the programming and debug interface is called “debug interface”.

3.5 General Design of Debug Interface

The debug interface consists of two different parts. The first one is the physical interface, which is often a standard communication interface, e.g. SPI, UART or even JTAG. The second part is a communication protocol, which is device and/or vendor specific.

The communication interface to outside, which means to the debug host, is a serial interface typically. To save pins a parallel interface is avoided. UART, IIC and SPI are common interfaces to implement a low pin debug and programming interface. The data rates are low so that no expensive high speed serial interfaces like PCI express are necessary. Often the target frequency is between 10 and 100 MHz.

For a small 16 bit microcontroller, like the openMPS430 [24], a communication protocol is sufficient, which contains a small subset of control words and set of 16 bit registers for address and data. For a larger processor, e.g., LEON2 [2], the size of the register has to be adapted and the complex pipeline requires more control information. Companies like Atmel, Texas Instruments or in projects from public domain projects like openMSP430 from openCores use own communication protocols, on top of physical interface, specific to their requirements.

During a scan chain test, the ASIC can be stopped after one clock cycle and intermediate results can be read out. The debug interface does not allow an access to the register at any clock cycle. A processor executes an instruction, which requires usually more than one clock cycle. After this, the debug interface allows a snapshot of all registers of the processor. An uncontrollable access to a device is a backdoor for an attacker, because a debug unit is fully integrated into the system so that every register and memory area can be read or written.

3.5.1 Device-specific Debug Interface

Figure 43 and 44 show different possibilities to integrate a debug interface into a microcontroller without a JTAG interface.

The LEON2 processor in Figure 43 [2] has an AMBA bus system and the debug unit consists of two components. First the debug serial link, which provides the UART interface, and second the debug unit itself. The debug unit is plugged onto the AHB bus and every data which is transferred on the bus can be captured by the debug unit and stored into a trace buffer.

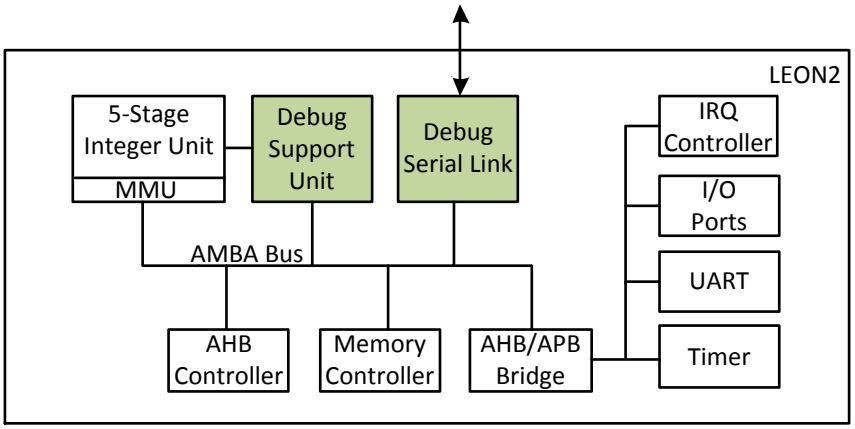


Figure 43: LEON2 processor with AMBA bus sub system and connected debug support unit and debug serial interface, marked in green. Debug interface and debug unit are separate units and the communication between them is done via the AMBA bus. The debug unit has access to the processor core and to the AMBA bus. The integrated MMU does not protect any memory area during accesses from the debug interface. Figure reprinted from [2].

The 16 bit microcontroller openMSP430 in Figure 44 [24] has an embedded debug unit, which is fully integrated and directly connected to the execution logic, the front end and the memory bus. For communication to the host an UART interface with its own protocol is used. The debug unit can send a “CPU halt” command, so that the program execution is stopped after the current instruction. Furthermore a read and write access to all memory areas including the CPU register is possible on-the-fly. This means that it is not required to stop the CPU for such operations.

3.5.2 JTAG based Debug Interface

The JTAG Standard 1149.1 [32] defines a serial interface. As part of the interface description the internal command structure is defined. In addition to test commands also a set of free usable commands are available. These vendor-specific commands can be used to implement a debug and programming interface.

As described in Section 2.3.2 a microcontroller from Atmel [3] uses a scan chain to get access to all registers at the serial debug interface. The interface itself follows the JTAG standard.

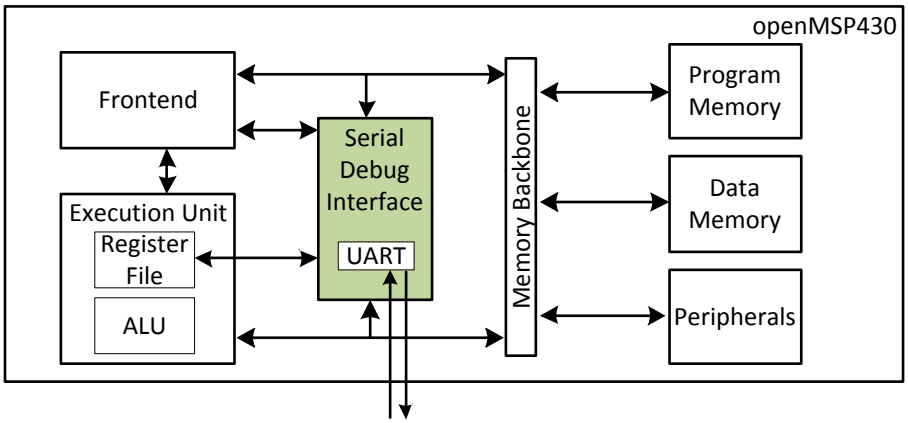


Figure 44: Debug unit, marked in green, in an openMSP430. It is deeply integrated into the system and it has access to all components. In contrast to the debug unit in the LEON2 processor, it cannot be removed. Figure reprinted from [24].

3.6 Secure Debug Interfaces

The misuse of a debug interface to attack a device is obvious. From a stolen device the stored program can be read out to get the program code or reprogramming of the device to use it for attacker's purpose. The security mechanisms are not so manifold for the debug interface as for the scan chain interface. Fuse bits or passwords are common countermeasures for attacks against debug interfaces. Special solutions, like the Mirror Key Register (MKR) as protection for a secret key register, are not meaningful.

3.6.1 Boot Loader for Password Protection

A boot loader can be helpful in different application scenarios. It allows a programming without a special programmer or additional features can be implemented subsequently without a programmer but with an interface connection to UART or USB.

In microcontrollers with a boot loader mechanism the non-volatile memory is divided into two parts and every part contains an executable program as shown in Figure 45. The first part contains the boot loader; the second part contains the application itself. The booting address is hard coded or user-defined e.g. by fuse bits. The second solution has the advantage that the image size of the boot loader and the application are flexible. Furthermore for a final programming of sensor nodes in a shipment state the boot loader can be removed.

After a reset the boot loader software is executed. A user interaction to interrupt the start of the application can be pushing of a button or a predefined command on a communication channel (e.g. 0x55AA on TX line of UART interface). If no user interaction happens after a certain time the boot loader jumps to the specific address of the application. Now the application is executed.

The range of functionality of a boot loader depends on requirements, available memory and the type of the microcontroller. Typically the NVM, which can be used by the boot loader, is in the range of several kBytes. This restricts the software routines to a minimum subset, e.g. implementation of a communication protocol via SPI or UART, simple authentication mechanism [82], or even a wireless update communication protocol [79], or the implementation of code verification.

An example for a password protection of the debug port is the MSP430 from Texas Instruments. To enable the access to the debug port a shared secret key, the password, has to be applied to the device by the user. The Bootstrap Loader (BSL) offers debug functionality and it is used to program an MSP430 device. But in addition to a standard boot loader as shown in Figure 45 the BSL is located in a secured memory area.

Every shipped device has a preprogrammed BSL which supports UART as serial communication link. If required the user is able to update the BSL code, e.g. to replace the UART protocol by an IIC protocol. Two different types of commands are supported: unprotected commands and password protected commands. Clearly the "password transmit" command is unprotected, furthermore the change of the baud rate and the full erase of the device. After sending the correct password to the MSP430 the full access to the device is enabled, the protected commands like reading and writing of the memory or setting the program counter can be executed. The password has a length of 256 bit, so that a brute force attack is nearly impossible. Later versions of BSL have advanced security features which can be set in a separate section of NVM. The BSL can be disabled fully or the content of the memory is deleted if a wrong password was sent to the device.

Due to the fact that the mass erase command in BSL is not protected by a password and the password is set to a default value the device can be fully used after the erase step. Furthermore the configuration of the BSL can be set into a state which initiates the deletion of content in the flash memory if the entered password is wrong. It cannot be avoided that the owner of the device fails during entering the password. But in this case the content of the device is deleted. This can be annoying because of the password is very long and a typing error is not unusual. The firmware of the BSL can be updated by the user. On one hand this can be an advantage to fix bugs or improve security features. On the other hand is can be a disadvantage in case that a programmer adds some bugs which are usable to get secret information from the device.

3.6.2 Authentication Mechanism

The password protection from Section 3.6.1 is a very simple type of an authentication mechanism. The authentication is based on a shared secret key. A trusted group of users knows the secret key, which is a very insecure type of an authentication in principle as it cannot be guaranteed that the secret key is kept secret.

A more secure solution for an authentication mechanism is a public-private key scheme. The protocol handling is done in software which runs on the system processor or in a specific hardware module. The simplest approach is a non-server based authentication. The public key is stored in the device and the private key is known by the application developer. Unfortunately the same weak spot like for password check is existing. The security depends of the private key being secret. Especially if many people use the same key, the risk exists that the private key is

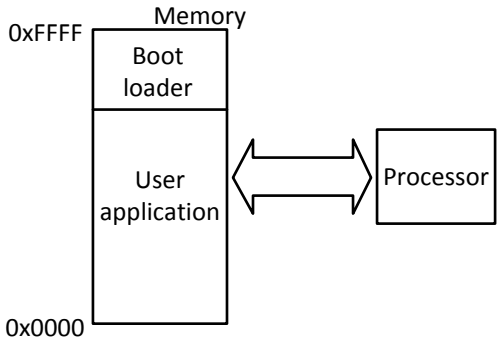


Figure 45: Boot loader mechanism of a microcontroller. Boot loader and user application are in the same NVM. After a reset the program from the boot loader section is executed. Without any event at the debug interface in a defined time slot the user application is started.

misused. A possible countermeasure is a server based authentication mechanism, which requires a connection to the server at least for the authentication step. The server is in the same network or it is reachable per internet connection. In the age of WiFi and UMTS it can be assumed that such a connection is available at most places.

A disadvantage of the authentication mechanism is the type of protocol handling. To establish a debug session the running application has to be interrupted and an application, which is responsible for the authentication, has to be started on the processor. This means that the running program cannot be stalled and debugged in the current state. Especially if the application program has an error, which causes an endless loop, a reset is required. In this case the information of the current program state, like value of program counter or content of the register file, is lost. A debugging is not possible until the error occurs again.

For an update of the user application, the authentication mechanism is well usable. The application has to be restarted after the update, so that an additional interrupt of the application to perform the authentication protocol is possible.

A further example for an authentication based debug port for WSN is described in [39]. The microcontroller is enhanced by an additional microcontroller with an RFID tag as shown in Figure 46. Both microcontrollers have independent power supplies. The microcontroller of the sensor node has its own power supply, e.g. a battery, but the microcontroller for debug does not have any permanent power supply. In case of debug request, the user has a RFID read/write module. This module provides electrical energy via its antenna. This amount of energy is sufficient to perform the wake up sequence. Afterwards the authentication scheme is done. In case of granted access the data is transferred per Bluetooth.

3.6.3 Destroying

In case of a dedicated debug and programming interface a simple approach of protection is destroying of the interface. On the input pad of the interface a high input voltage is supplied which

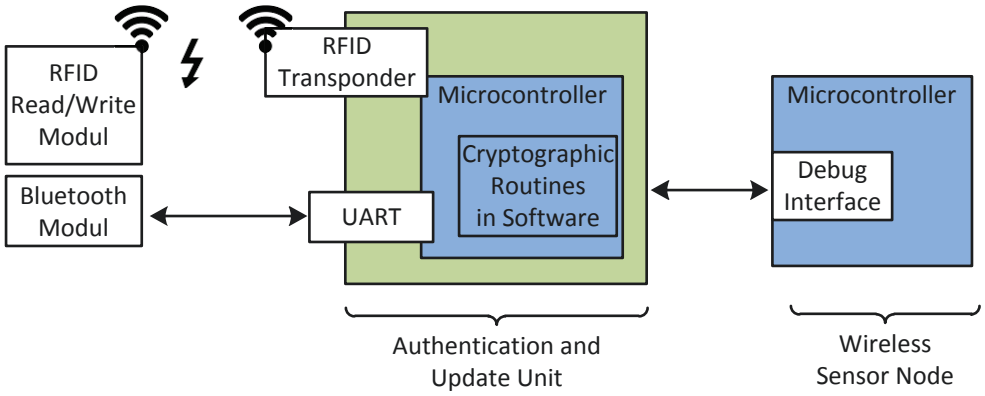


Figure 46: Wireless authentication approach for debugging. A second microcontroller performs the protocol handling. Figure reprinted from [39].

is much higher than the breakdown voltage. E.g. in the IHP $0.25\,\mu\mathrm{m}$ technology the maximum breakdown voltage is specified as $6\,\mathrm{V}$ for an nmos transistor. This means a voltage higher than this breakdown voltage destroys a transistor. In case that the debug interface uses some shared pins, e.g. on GPIOs, it is not possible to destroy the interface in that way. Furthermore the approach is dangerous because it cannot be guaranteed that no other circuits inside of the ASIC are affected. Figure 47 shows a photograph of destroyed input pads. Although both pads are destroyed under the same conditions, this means the applied voltage and current were identical, the result is different. It shows that applying a high voltage causes unpredictable destroying of semiconductor structures.

With the knowledge of the risk this solution can be used for a handful of devices in very early prototype phase of a project. Keep in mind that the interface is not usable anymore and a reprogramming of the internal program memory is not possible.

3.6.4 Fuse Bits

A fuse bit is a security mechanism, which is based on applying a high voltage to destroy a semiconductor structure in a well-defined way. It can be implemented in different ways as described in Section 2.5.2. A very common implementation is the anti-fuse. After device programming and debugging on the gate oxide of a transistor a high voltage is applied which causes that the oxide is destroyed. For example the Cortex processor in [72] contains three fuse bit banks, each 255 bit. Since bank 1 and 2 are for user-defined data, bank three contains control bits and 4 bits to disable the JTAG interface. Once the interface is switched off, it cannot be enabled without high effort. The die has to be decapsulated and in a FIB the broken connection has to be repaired. Such equipment is very expensive and a well-educated engineer is required. But companies offer the usage of a FIB as a service for external customers, which reduces cost of such an attack dramatically.

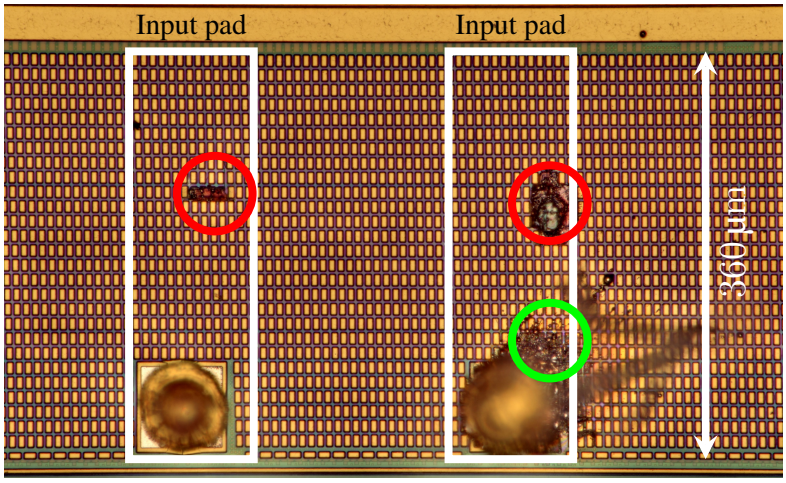


Figure 47: Photography of an ASIC in IHP $0.25\,\mu m$ technology. Two input pads of a microcontroller which are destroyed by a high input voltage. The transistors are covered by technology fillers, so they are not visible. Although both pads are destroyed under same conditions, the result is different. In the pad on the left side only a small area seems to be affected (red ellipse). This is the internal driver structure. The pad on the right side is damaged first near the bond wire (green ellipse) and second inside of the driver structure (red ellipse) too.

The MSP430 family offers a second debug interface at the JTAG port. This interface is protected by "electronic fuses". In contrast to a fuse in OTP manner such a fuse is resettable. The test interface is enhanced by several additional registers, which is compliant to the IEEE 1149.1 standard [32]. Once the fuses are blown an access to debug the device is only possible using the BSL and with a command the fuses are resettable. The JTAG port as additional debug port offers the possibility to test and debug the device on the PCB. This is useful if more than one JTAG enabled device is mounted on PCB to test the connection between the devices. But the user has to keep in mind that two debug interfaces have to be disabled to avoid unrestricted access.

The difference between BSL and JTAG based password protection is the check of the password. While BSL is a software solution with a type of a boot loader, the JTAG based protection mechanism is a pure hardware based approach. In case of BSL the MSP430 processor has to stop the execution of the application and to perform the password check.

3.6.5 Module Based Debug Unit

The misuse of the debug interface can be avoided if the debug interface is not part of the final ASIC. In other words: the ASIC is fabricated twice. One version with the debug interface and one version without the debug interface. In case that the debug interface is built as a separate layout macro, as shown in Figure 48, it can be removed very easily. Nevertheless two different mask sets are necessary.

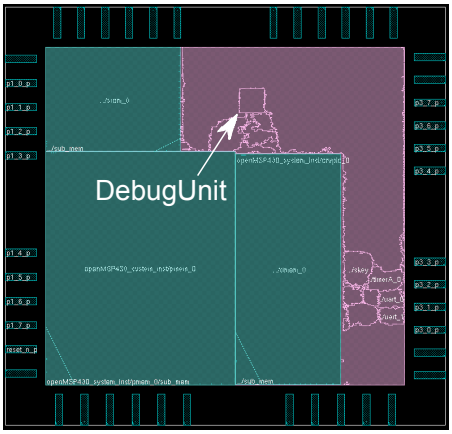


Figure 48: The debug unit is integrated as a hard macro so that it can be easily removed for a version without a debug unit.

The second approach is the fabricating of the device without debug interface. The software development is done on an FPGA implementation, which contains the debug interface. Analogue components like radio or an ADC cannot be mapped into FPGA resources. So it is not possible to test software routines, which use these components.

The debug interface is typically used to store the program in the internal non-volatile memory, but for both ideas the problem exists that the device cannot be programmed. It can be used for microcontroller with ROM. The disadvantages are that the program has to be ready before the final version of the ASIC is produced and an update is not possible. So these approaches are not suitable in area of fully integrated microcontrollers, which have an internal reprogrammable memory.

3.7 Attacks on Secure Debug Interfaces

Clearly unprotected as well as protected debug interfaces are in focus of attackers. In an unprotected device the internal data are accessible for everyone. A suitable programmer and free available software are sufficient, to read out and reprogram the attacked device.

Access to a device, which is protected, is more difficult. In [25] an attack against the boot loader mechanism of the MSP430 is described. The password to enable the debug functionality has a length of 256 bit. This is equivalent to 32 Bytes. A brute-force attack for such a key is impossible. But the software routine of the boot loader checks the password byte by byte. In case of a correct byte of the password, the comparison is two clock cycles faster than in case of a wrong byte. Since this difference in program execution is observable from outside, each byte of the password can be tested individually. As result after 32x256 tests the correct password is determined by the attacker.

Interfaces which are secured by fuse bits can be enabled, but with a higher effort. As described in Section 3.3.12 it is possible to repair a blown fuse bit. Clearly the effort and cost are too high to steal a sensor node and to reprogram the device for attacker's purpose. In fact repairing of a

fuse bit is interesting, if the device contains secret data or the software itself which can be reused by the attacker.

3.8 Discussion

In principle two different types of protection mechanisms exist. On the one hand in an irreversible process the interface is destroyed. Once the interface is destroyed a debugging and reprogramming is not possible anymore. Without opening the package and repairing the structure with a FIB the access to the internal of the device is not possible.

Table 2: Countermeasures for programming and debug interface and their features and properties. The evaluation based on estimation of the author of this thesis.

Countermeasure	*Design independent*	*Technology independent*	*Implementation effort*	*Reverse engineering possible*	*Area overhead*	*Additional time to init debug*	*Secure against attacker level*	*Probing attack possible?*[1]	*Leakage problem?*	*Glitch attack*	*Reset required before debugging?*	*After reset further debug possible?*	*Multiple usable?*
Boot loader (BSL)	no	yes	–	no	-	yes	1	no	yes	yes	yes	no	yes
RFID based authentication	yes	yes	–	no	–	yes	1	no	no	no	no	no	yes
Physical destroying (pad)	yes	yes	++	no	++	no	1	yes	no	no	no	yes	no
Fuse bits	yes	yes	++	no	++	no	1	yes	no	no	no	yes	no
Fuse bits (electronic)	yes	yes	++	no	++	no	1	yes	no	no	no	yes	yes
Module based Debug Unit	no	yes	++	no	++	no	3	no	no	no	no	yes	yes

Legend:
++ very good + good o average - bad – very bad

[1] Influence system with needle probe, e.g. force logic "1" on a specific wire

On the other hand a type of authentication process, this allows an access at any time. The chosen approach depends on the selected system, application area and security level. The debug interface is used after the post-production test of the device. It means that the error free device can execute some software, e.g. a boot loader. The boot loader is a small piece of software with a restricted number of functions for protocol handling and authentication. Since the device is fully tested, there is no risk that a production error enables the interface accidentally.

Table 2 gives an overview of countermeasures for debug and programming interface. The development team of a microcontroller has to consider the type of security mechanism in an early stage of the project. While the fuse bits can be integrated with a low effort into a design,

the boot loader mechanism requires additional effort for development. Once the fuse bit is blown it cannot be reverted and a device with a software error is lost. But the device with the boot loader can be updated.

In sum the ideal debug and programming interface does not exist. Each approach has its advantages and disadvantages. Even the destroying of the pads can be ideal solution, because it is cheap and it works at any device with separate pins for the interface. A solution is desirable, which allows a debugging and reprogramming in several stages, but even with the option to fully disable those features.

4 Secure Scan Chain Interface

In the following chapter the solution for a secure scan chain interface (SSCI) developed in this thesis is shown. At first the requirements are summarized, followed by a sketch and a detailed description of the proposed solution. The integration into the protected system is described including an evaluation and the description of the modified post-production test. A complex security analysis and alternative implementations are shown, too. At the end the patent situation is explained and a conclusion closes this chapter.

4.1 Requirements

The precise goal of this work is a protection mechanism, which allows a full and complex post-production test and that prevents any unrestricted access which can be misused for scan attacks. At a first glance, a BIST is a suitable solution for enabling a secure test. However, it provides only limited fault diagnosis capabilities. Such an extensive diagnosis is indispensable to improve the production rate.

The access to the internal data of a device, which may include secret data, should be completely restricted to authorized users only. The integration of the security component should be seamless and in any case compatible to standard interfaces like JTAG and any other proprietary test interfaces. Especially the generation of test pattern should be done by the standard tools like TetraMax (Synopsys) [78] or Encounter DFT Architect (Cadence) [9]. Any modification of test pattern is prone to errors and requires additional verification. The influence on additional area and power should be as low as possible. Even the path delay should not be highly increased.

The progress in technology development causes a permanent scaling of transistors and metallization layers. Shrinking a purely digital design to a smaller technology is a fast process compared to an analogue design. Thus, the security feature should be implemented in a way that a porting to a smaller technology does not require high effort. This means that the solution and most of its components should be scalable in an automatic way.

A generic approach is required so that the solution is not restricted to a special design or IP core. It should be usable for any design, independent of its function. After fabrication and packaging the device is typically soldered onto a PCB. Often such a PCB has several ASICs mounted and they are testable with a JTAG chain. The solution shall be compatible with typical scan interfaces and support such board level tests.

In principle a secure scan and a secure debug interface have similar requirements (see Section 5.1), so that an approach for a secure scan chain interface is possibly reusable for a secure debug interface.

In summary, the requirements for a secure scan chain interface are:

- Secure against access by unauthorized users
- Test pattern generation with standard tools
- No restriction on fault diagnostic
- Compatible with standard test equipment
- Minor influences on the timing
- Small area overhead

- Low power consumption in functional mode
- Adaptable for future technologies
- Generic approach
- Applicable for standard interfaces like JTAG

As shown above, the requirements are manifold and partially in contrast to each other. The greatest challenge is to combine security and fault diagnosis.

4.2 Concept for Secure Test

A scan chain is a possible solution to fulfill the requirements, if it is enhanced by a strong security mechanism. The core idea for the security mechanism is based on a matching key pair. Figure 49 shows the general system architecture of approach proposed in this thesis. The core to protect is surrounded by the Secure Scan Chain Interface (SSCI), which is implemented as wrapper component. The SSCI protects the core by disabling the `scan_enable`, `scan_in` and `scan_out` signal after fabrication, to hinder a misuse of the scan interface directly after shipment of the die. After finalizing the post-production test the core is protected too.

The security controller is responsible to write and delete a secret key in the OTP. The "golden key and compare unit" is instantiated multiply times in the devices and every unit contains one bit of the golden key. In case the correct secret key is stored in the OTP the `scan_enable` signal is forwarded from the pad to the core and the `scan_in` and `scan_out` signal are unlocked.

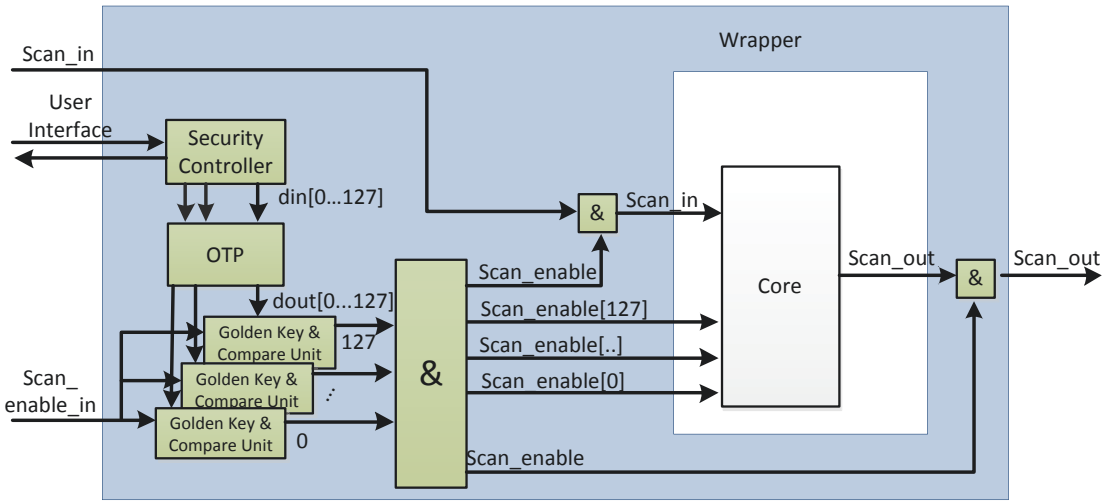


Figure 49: Detailed sketch of the secure scan system. A wrapper around the core contains the security mechanism for the access control. Although the wrapper looks large and complex, the implementation requires less than $0.14\,\mathrm{mm}^2$ in a $0.25\,\mu\mathrm{m}$ technology. The core is modified slightly. The single bit input of `scan_enable_in` signal is replaced by a 128 bit bus. This bus and the scan data input and output are protected by the security mechanism.

The system is used in the following way: First, the golden key is stored in a hidden way within the device. The key is fixed and embedded during ASIC production. Second the secret key has to be written by the test engineer into the device. If both keys are matching, the scan chain interface is enabled. As long as the secret key is not correct the `scan_in` and `scan_enable` signals are not forwarded to the core and the output of the `scan_out` signal is blocked.

The golden key is part of the design and it is stored in a set of compare units which are resistant against reverse engineering. The secret key is stored into an OTP. Once the secret key is deleted, a FIB and a very high effort are required to re-enable the scan interface again. This reduces the number of potential attackers significantly. Furthermore, in contrast to approaches with only a single fuse bit, a multi-bit entry in an OTP, the key, is used. Consequently, an attacker has to fix a high number of blown fuse bits. A simple modification of every bit in the OTP is not sufficient. Instead, the correct secret key has to be set. This is a complex task without very detailed reverse engineering. The key has to be determined in brute-force manner which is nearly impossible for attackers of class 1 to 3 and even very difficult for attackers of class 4. A brute-force attack means that in worst case 2^{128} different secret keys have to be tested until an access via scan chain is possible.

4.3 Memory for SSCI and SDI

4.3.1 Requirements

The secure scan and debug interfaces have their own requirements regarding memory. Two different memories are required; one memory for the golden key, the second memory for the secret key. The size of each memory does not exceed 128 bit for the key, plus some control bits.

The golden key and the secret key are identical and they have to be keep secret. The golden key should be permanent which means that an attacker is not able to change its value. Furthermore reverse engineering should be very difficult, and in the best case, the golden key can be set individually during the production step, e.g. for a group of ASICs without additional costs or high time effort. The required silicon area of the security feature should be small, but for additional security it is allowed to increase the area.

The read time of all memory types is fast enough to implement scan and debug interface which operate at a low frequency. The time to write is not important. The amount of data, which is written, is very small, so that the required time to write is short. During the testing and debugging a higher power consumption is allowed even though the device's power consumption should be as low as possible during operation mode. Most devices are battery driven, so that additional power consumption reduces their life time.

4.3.2 Golden Key

The value of the golden key should be fixed for the lifetime of the device. With a re-writable memory, an attacker would be able to set the value. A setting of the key in groups should be a possible feature to avoid the same key for different customers. The memory for the golden key should be difficult to identify in layout, and it should be even more difficult to extract the content of the memory in a reverse engineering process. Otherwise, it is possible for an attacker to read out the value and set the "secret key" to same value. That is why a direct reading access to the memory, where the golden key is located, must be forbidden.

4.3.3 Secret Key

The secret key must be programmable and erasable by the test engineer for the post-production test or by the application developer for a debug session. Furthermore, the memory should be either a slow writable memory or one-time writable memory.

From the view of a test engineer, a slow writable memory is a negative requirement at first glance, because it increases the time for testing. A long time, which is required to write the value to the memory, is helpful against brute-force-attacks. In brute-force-attacks, all possible keys is tested in a trial and error approach to identify the correct secret key. Statistically in a brute-force-attack the correct key is found after testing half of all possible key. For example a flip flop is writable in a fast manner, so that a lot of keys can be checked in a short time. Possible countermeasures against brute-force-attacks are the use of a long key, or a long writing-time for the key, or a restricted number of write operations or combinations of them.

The scan interface is typically used only once for the post production test, so for this memory one write operation should be sufficient.

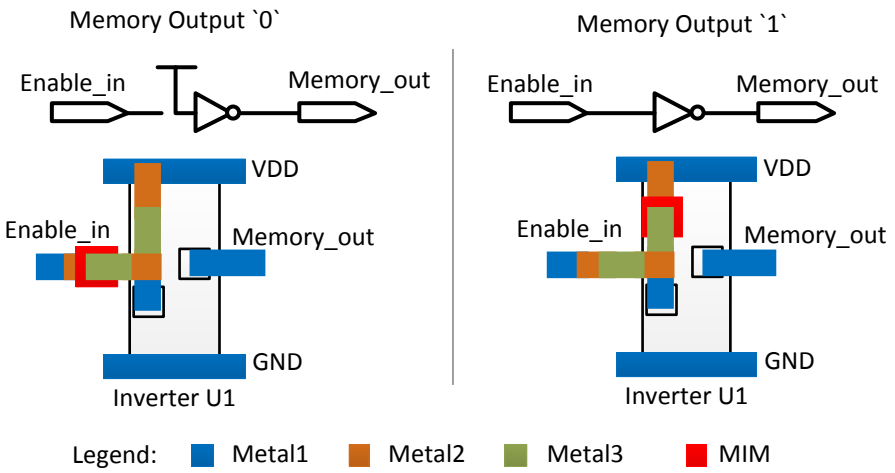


Figure 50: MIM-based ROM with enable input. The cell on the left side with output “0” is connected to VDD permanently; the cell on the right side with output “1” has the function of a inverter. To improve the visibility the MIM structure is shown larger.

If different development cycles require several debug and programming steps and the memory is writable only once, the memory should offer several entries, 128 bit each.

4.3.4 Introduction of MIM-based ROM

The MIM-based ROM was developed as part of this work and it is a special type of combinational logic ROM. It bases on digital standard cells and resists optical reverse engineering. In contrast to a pure digital design an additional layer, during fabrication, is used. The MIM layer is typically used for mixed-signals designs or pure analog circuits.

In Figure 50 one possible implementation of such a ROM is shown. It contains one stored bit. The value is placed at the output when the “read enable” signal is set. The difference between Figure 50(a) and Figure 50(b) is the output value. Once it is zero and once the output value is one. It is controlled by a MIM structure, which is marked in red in the figure. During fabrication the MIM layer is produced as very thin isolation layer (58 nm), made of silicon nitride. In a reverse engineering process, using an optical microscope, it is not possible to identify the stored bit value. Like every other type of ROM the value is fixed in the layout phase.

The example bases on modification of digital standard cells and their wiring. The implementation on an array structure like Section 2.5.2 is also feasible.

4.3.5 Evaluation

In Table 3 the properties of different memory types are summarized with respect to the application area of a secure scan and debug interface. The requirements, of the described secure scan

and debug interface in this thesis, are almost the same for both interfaces and are described in Section 4.3.1. In Table 3, additional properties of the memories are considered, which are not relevant in most application areas, but to secure scan and debug interfaces, they are important. For example, a programming after production simplifies the requirement of different keys for different customers. The possibility of reprogramming, or reading of the value in an optical way, or changing of the value (e.g. in a laser attack), is a weakness.

Table 3: Benchmarking of different memory types for 128 bit memory for storage of a secret key.

Memory type		*Programming after production*	*Groupwise programming ASIC*	*Reprogramming*	*Area per bit*	*Timing (read)*	*Timing (write)*	*Power*	*Distributed in layout?*	*Resistance against reverse engineering*	*Resistance against laser attack*	*Implementation effort*
VM	Flip Flop	yes	n/a	yes	+	+	++	-	yes	++	-	++
	SRAM	yes	n/a	yes	-	+	++	+	no	++	–	–
NVM	combinational logic ROM	no	no	no	++	++	n/a	o	yes	+	++	++
	Array ROM	no	no	no	o	++	n/a	o	no	-	++	o
	MIM	no	yes	no	++	++	n/a	+	yes	++	++	+
	Laser fuse OTP	yes	yes	no	-	++	++	++	no	–	–	+
	Electrical fuse OTP	yes	yes	no	-	++	++	++	no	–	–	-
	Anti-fuse OTP	yes	yes	no	-	++	–	++	no	++	++	-
	EPROM	yes	yes	yes	–	+	+	+	no	++	–	–
	Flash	yes	yes	yes	–	+	+	+	no	++	–	–
	RRAM	yes	yes	yes	+	++	–	++	no	++	++	–
	SRAM based PUF	no	no	no	+	–	n/a	+	no	++	–	–

Legend:
++ very good + good o average - bad – very bad

The silicon area of the memory is taken into account for a memory which has 128 bit plus some additional control bits. Additional area increases the cost, so that is getting more difficult to establish a secure scan and debug interface.

In general it is possible, to use NVM for the golden key as well as for the secret key. Every memory which is reprogrammable is not usable for the golden key without additional counter-

measures. Otherwise an attacker is able to set the value of the golden key by himself. The attacker's golden key can be written into the secret key memory and the access to the interface is not protected anymore.

All memories which are writable after production have an additional weak point. An attacker would be able to program all bits with a known value, if he gets a untested ("raw") device. The secret key can be set to the same value and there is no security anymore for the interfaces. An additional countermeasure is required to prevent this type of attack.

The timing for read is not important because the scan as well as the debug interface work at a low frequency so that the timing for read is not a restrictive factor. For a write operation the required time does not matter, moreover a very slow write time has the advantage that a brute-force-attack requires more time. This is the reason why memories with long write time get a better rating in Table 3 than memories with short write time.

In area of WSN the power consumption is an important fact. Each additional component reduces the lifetime of the battery and so the lifetime of the device. For example for volatile memory a flip flop consumes more energy to store a bit value than an SRAM. For non-volatile memory a very energy efficient solution is the laser fuse OTP. A bit value is stored in type of a cut or uncut wire without any transistors. So this type of memory has zero leakage.

Another application issue is the structure and the layout of the memory. A flip flop based memory is made of standard cells which are arranged in the standard cell grid. It is difficult to identify them and to grab signals. For non-volatile memory the combinational logic is a possible implementation for memory which is hard to identify in an ASIC. Memories which are built as blocks are clearly visible in an ASIC layout. As example different types of memories are marked in Figure 7. They cannot be hidden in layout and a probing attack might be easier. Furthermore some types of memories allow a determination of the content by visible inspection. In Figure 13a an example is given for such a memory in whose the content can be read via microscope. A fully contrast to this block based memories, the MIM based memory is distributed in layout and, even in a typical reverse engineering step, the content of the memory cannot be extracted.

A typical type of attack against memory is a laser attack, in which the content of the memory can be read out or even modified. Such an attack requires an opened package in a manner that the device is still working. A laser fuse can be set very easily, because this type of memory was developed for programming it by a laser. Furthermore the electrical fuse OTP seems to be vulnerable for this type of attack. A short metal wire looks similar to a laser fuse OTP, but it is programmed by a high current. It might be possible, that this piece of metal can be cut by laser in the same manner as for the laser fuse OTP.

The content of the flash or EPROM can be influenced by an attacker via a laser. Anti-fuse OTP and a memory in RRAM technology are resistant against read out or modification. The data is stored in a way, which cannot be influenced by a laser.

A last important fact for selection of the memory type, is the implementation effort. For example, flip flops are part of every digital standard cell library. For an array ROM a simple memory generator is required. For an SRAM such a generator is quite more complex and for flash or RRAM a very high effort is required in development of the technology and in implementation of the generator.

Table 3 describes different properties of each memory type, Table 4 contains a list of all possible combinations for golden and secret key. For golden key only non-volatile memory is used, because the nature of this key is that the value needs to be stored even after power-off. For the secret key both types of memories are possible, volatile and non-volatile memory.

A laser fuse OTP for the secret key is not usable in any case, independent of the used memory type for the golden key. An open package would be required to program the secret key. The cost and the effort are too high for such a solution. The electric fuse OTP has the disadvantage that the usage for golden and/or secret key is very risky from a security point of view. The fuse element is relatively large and the content can be read out using an optical microscope.

In contrast to the electrical fuse OTP the anti-fuse OTP is well usable for the secret key. It can be written only once. Nevertheless it is possible to delete the entry without additional countermeasures. The storage element is the gate oxide of a transistor and the value cannot be determined without high effort (usage of FIB).

EPROM and flash based memory are usable in case of short secret keys. The limited number of write cycles and the slow time to write prevents a brute-force attack. But RRAM, and especially flip flops and SRAM, offer a high number of write cycles in a short time, which might be enables a trial and error until the secret key is found.

For the golden key all types of non-volatile memories are usable. Most of them are not well usable or they are unsecure. From security point of view the anti-fuse OTP and the RRAM are resistant against attacks. Unfortunately the RRAM technology requires a very high effort in production and it is available from selected manufactures only. Nevertheless these types of memories are built in as complete memory blocks, so it is possible to identify them in a layout. With low effort the outputs are identified and in a further modification step in a FIB the outputs can be set to a defined value.

4.3.6 Discussion

At the first view a volatile memory is the best choice for the secret key. After power-off the memory is empty and the corresponding function is not enabled as before. The volatile memory has two disadvantages. The first one is the fast reprogramming, which is possible. If the secret key is short a brute-force-attack can be successful. Clearly this disadvantage can be solved by an additional countermeasure. The second disadvantage is that a key is writable at any time and it is not possible to deactivate writing keys. This means that the interface can be enabled at any time during the lifetime of the device.

In summary no combination of different, traditional memory types in Table 4 is very well suitable for the SSCI as well the SDI. Since the secret key can be stored into an anti-fuse OTP, the golden key cannot be stored in a secret way. All available types of memory do not fulfill the requirements, so a new method is necessary to store the golden key. The new introduced MIM-based ROM is well usable and a secure solution.

Table 4: The ratings for all combinations of memory types for golden and secret key are shown.

Secret key: / *Golden key:*	*Laser fuse OTP*	*Electrical fuse OTP*	*EPROM*	*Flash*	*RRAM*	*Flip Flop*	*SRAM*	*Anti-fuse OTP*
combinational logic ROM	–	-	o	o	-	-	-	o
Array ROM	–	-	-	-	-	-	-	-
MIM-based ROM	–	-	+	+	o	o	o	++
Laser fuse OTP	–	-	-	-	-	-	-	-
Electrical fuse OTP	–	-	-	-	-	-	-	-
Anti-fuse OTP	–	-	+	+	-	-	-	+
Logic EPROM	–	-	+	+	-	-	-	+
Flash	–	-	+	+	-	-	-	+
RRAM	–	-	o	o	-	-	-	o
PUF	–	-	+	+	-	-	-	+

Legend:
++ very good + good o average - bad – very bad

4.4 Description of Secure Scan Chain Interface

In the following the technical details of the SSCI are described. All components, which are required for the SSCI are combined in a wrapper and the protected core is embedded into this wrapper component. The main focus is on the new developed compare units, CMP0 and CMP1, which are resistant against optical reverse engineering. In the following the OTP as a one-time key storage and the user interface are explained. To perform the key handling, a small controller is part of the system too. Finally the verification of the developed system is described. The SSCI is implemented in IHP $0.25\,\mu\mathrm{m}$ technology.

4.4.1 Compare Units

The concept of the compare units (CMP) is developed as a part of this thesis. These units are used as storage element of the golden key with an integrated comparison of the key input. Each compare unit contains one bit of the golden key and two different types are necessary: CMP0, for bit value zero and CMP1 for bit value one. A set of 128 compare units compose the golden key and compare the corresponding output, the secret key, from the OTP.

Schematic In Figure 51 and 52 schematics for both types of cells are shown. In one configuration the logical value "0" at the `key_in` input enables the output (CMP0), in the other configuration the logical value "1" does (CMP1). The approach bases on digital standard cells and a wiring between them. The new cell element consists of three logic gates: two inverters and one NAND gate.

The main feature of this approach is the resistance against optical reverse engineering. In contrast to a pure digital design an additional layer, during fabrication, is used. Usually the MIM layer is used for mixed-signal designs or pure analog circuits. The MIM module is used as capacitor in analogue circuits, so that it in the schematic it is shown as a capacitor with a capacitance of $1.6\,\mathrm{fF}$. For comparison the input capacitance of the used inverter is $2.6\,\mathrm{fF}$. This means that the MIM structure has less influence than an additional connected inverter.

The difference between CMP0 and CMP1 element is realized by placing a MIM structure at different positions. Figure 53 shows the approach of a MIM layer. The integration of the additional isolation layer into a digital signal path is used to control the property of the circuit. The inverter U1 is combined with a MIM structure and it is used to connect or disconnect the signal from the inverter. The inverter U2 and the NAND cell U3 finalize the circuit. These cells are equivalent to an AND gate and enable the output if the input is the expected value. The truth tables are given in Table 5 and Table 6.

The CMP units can be constructed as new elements of the standard cell library. Since these units are built from digital standard cells, the effort for implementation is small. This is an important fact, otherwise a time consuming process is required to build completely new cells.

Layout Between the different metal layers silicon dioxide is placed as isolation layer with a high of several hundred nanometers. The vertical interconnect between the metal layer is made by a via structure with a conductive material, e.g. titan nitride. The MIM structure is an additional very thin isolation layer ($58\,\mathrm{nm}$), made of silicon nitride and it is marked in red in

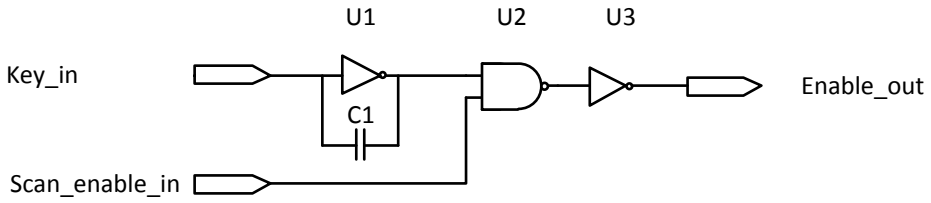


Figure 51: Schematic of CMP0. U1, U2 and U3 are digital standard cells and the capacitor C1 represents the MIM structure. The `key_in` signal is inverted by U1. U2 and U3 implement an AND gate, so this module activates the `enable_out`, if `key_in` signal is logical "zero".

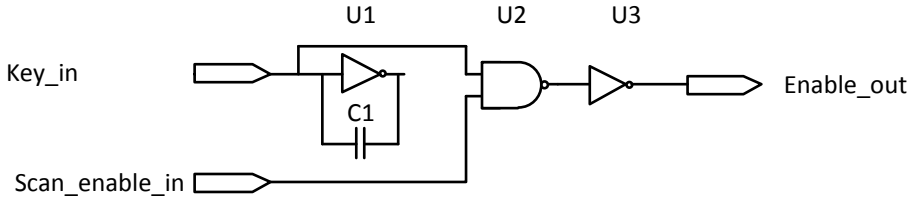


Figure 52: Schematic of CMP1. U1, U2 and U3 are digital standard cells and the capacitor C1 represents the MIM structure. In contrast to CMP0, the inverter U1 is bypassed and the unit enables the output signal `enable_out`, if `key_in` is logical "one".

Table 5: Truth table CMP0

Key_in	*Scan_enable_in*	*Enable_out*
0	0	0
0	1	1
1	0	0
1	1	0

Table 6: Truth table CMP1

Key_in	*Scan_enable_in*	*Enable_out*
0	0	0
0	1	0
1	0	0
1	1	1

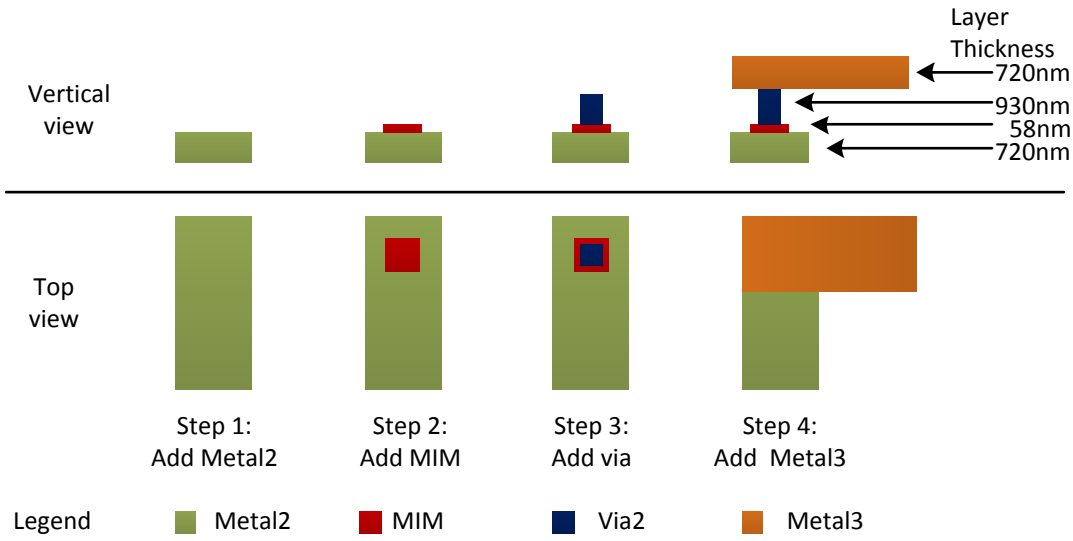


Figure 53: Flow of integrating the MIM structure between two metallization layers in vertical and top view. As shown in the top view, the MIM structure is not visible in an optical reverse engineering process.

Figure 53. It is placed between the metal and a via and the used in analogue designs to build on-chip capacitors. For fabrication this isolation layer requires an additional mask and each MIM structure is a rectangle on the mask. Based on this element the new CMP units are developed.

In Figure 54 an abstract view of CMP0 and CMP1 is shown. The digital standard cells are placed like in the schematic. The inverter U1 at the left side is responsible to invert the incoming key bit from the OTP, if the compare cell should test for logic “0” (CMP0). The output of the inverter U1 is connected to an AND logic, made of a NAND gate U2 and an inverter U3. It is necessary because of the used digital standard cell library does not contain an AND gate. This logic structure enables the secure `scan_enable` output if both inputs, inverted key and scan_enable, are “1”.

For the case that the compare cell (CMP1) should test on “1” at the input the MIM layer is placed as shown in Figure 54. In this layout the first inverter is bypassed so that the signal of the key bit is directly connected to the AND logic.

In Figure 55 a layout view, extracted from Cadence Virtuoso, of CMP0 (Figure 55a) and CMP1 (Figure 55b) is shown. A single standard cell has a wiring on metal 1 only. The compare units have a wiring on metal 1 to metal 3 and the MIM structure is placed between metal 2 and metal 3.

Simulation The simulation for the new developed compare units was done in different levels. At first the schematic, Figure 51 and 52, of the compare cells were simulated in SPICE. The design of the compare cells bases on digital standard cells. These cells are verified before the

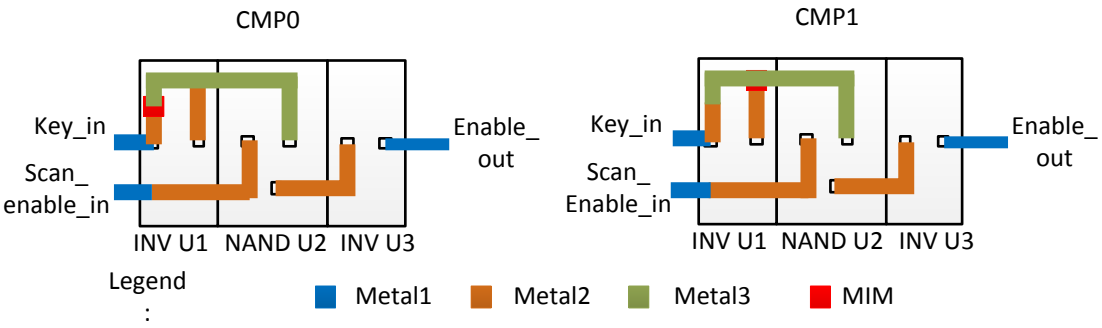
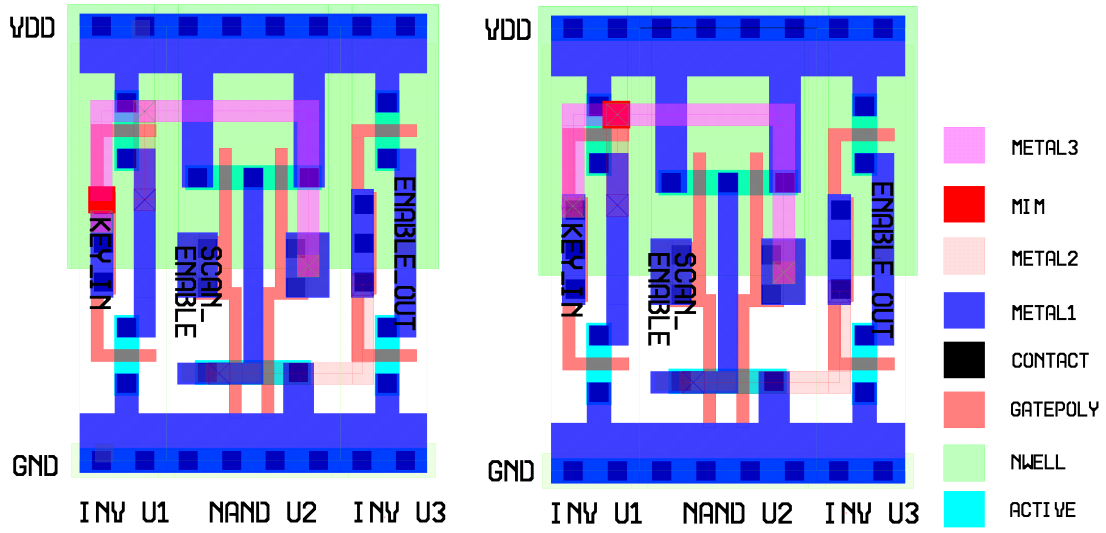


Figure 54: Abstract layout view of CMP0 and CMP1. CMP0 enables the output if `key_in` is logical "zero" and CMP1 if `key_in` is logical "one". The wiring of both units is identical and in an optical inspection no differences between both types are visible. A small, invisible MIM element at different positions is responsible to include the inverter U1 into the logic path for CMP0 or to exclude the inverter U1 for CMP1.



(a) Layout view of CMP0. The MIM element blocks the direct connection from `key_in` to the NAND gate.

(b) Layout view of CMP1. The MIM elements bypassed the inverter U1.

(c) Legend

Figure 55: Simplified layout of CMP0 and CMP1 in IHP 0.25 µm technology.

library is shipped, but the additional MIM capacitor may influence behavior and timing. The simulation in Figures 51 and 52 are done to verify the correctness of the functionality.

In Figures 56a and 57a a further simulation setup and its results are shown. To get the impact of the MIM capacitor the compare cells with capacitor and the modified compare cells without capacitor are simulated. At first the signal for the key input is set with a value which is able to enable the SSCI. This is shown in Figures 56b and 57b. After a short delay the `enable_in` signal changes the logical value from "0" to "1" at the position of Marker V1. At marker V2 the `enable_out` signal switches to "high". For CMP0 and CMP1 the internal delay is $0.09\,\mathrm{ns}$. The second simulation in 56c and 57c the `scan_enable` signal is fixed and the input for the key is changed at marker V1. The compare unit CMP0 without capacitor switches faster, at marker V2, in contrast to CMP0 with capacitor at marker V3 in Figure 56c. The influence of the additional capacitor causes an additional delay of $0.03\,\mathrm{ns}$ and this is marginal. For CMP1 is no difference between the schematic with and without capacitor. The timing is not influenced because the MIM is not in the signal path. The additional delay in Figure 56c is negligible. This is a situation which does not happen in normal operation mode. The output from the OTP which is the key input for the compare units is static.

Conclusion The schematic with the layout, as described above, is an example for an implementation of the compare logic. The key feature of the units CMP0 and CMP1 is that their layouts are almost identical. The only difference is the placement of a small and thin isolator at specific positions. This difference is invisible for optical inspections and makes the realization secure against reverse engineering. Depending on requirements and available digital standard cell library the schematic and layout may differ from the specific implementation sketched above. The influence of the MIM capacitor is negligible, in the worst case the additional delay is $0.03\,\mathrm{ns}$, and a remodeling of the behavior with respect to the timing is not necessary.

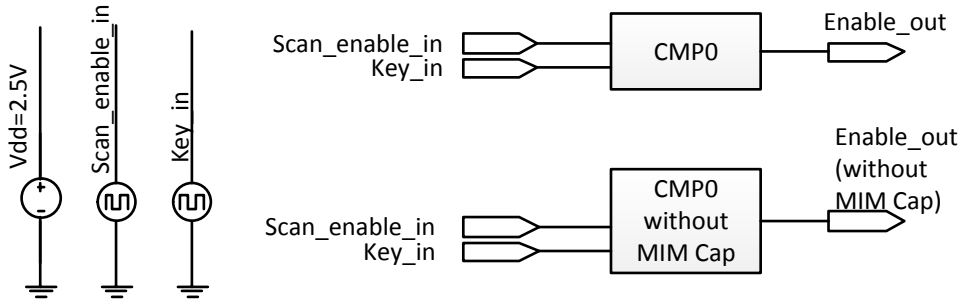
4.4.2 AND Module

In Figure 58 the connection between OTP, compare units and scan flip flops is shown. All compare units are connected to AND gates. The output of the AND gates is the `scan_enable` signal to the scan flip flops. This combination of signals is necessary to ensure that the scan chain is enabled only if all bits of the secret key are correct. Even the `scan_in` and `scan_out` port are protected by AND gates. To simplify the integration of this protection mechanism, the AND gates are implemented in one module combined with the compare units.

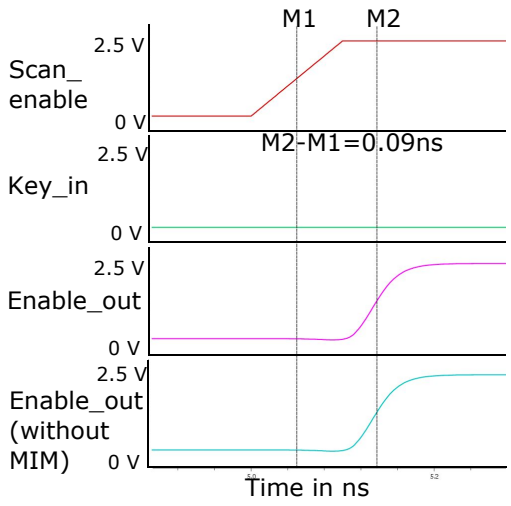
4.4.3 OTP as One Time Key Storage

Design Decision The golden key is hard-wired into the device, but the key to enable the scan chain has to be stored into the device on user request. This means that a memory is required, which stores the key during the scan chain test. In principle any type of memory can be used in this secure scan chain interface to store the secret key for enabling the scan chain.

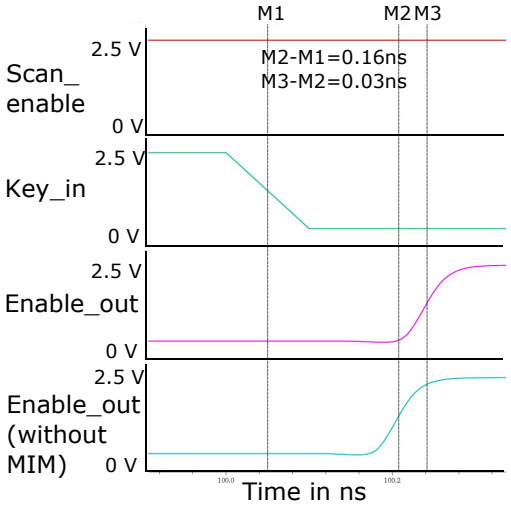
Typically a memory block like flip flops, RAM, flash or even the OTP can be written and read by the user. The time to write a data value into a memory depends on the memory type. A flip flop or a RAM typically have a fast write and rewrite time of less than 10 nanoseconds. RAM,



(a) Schematic for evaluation of CMP0. The security mechanism bases on a MIM capacitor, so the influence of the capacitor is simulated in this setup. The CMP0 unit is simulated twice. Once with the MIM capacitor, once without the MIM capacitor.

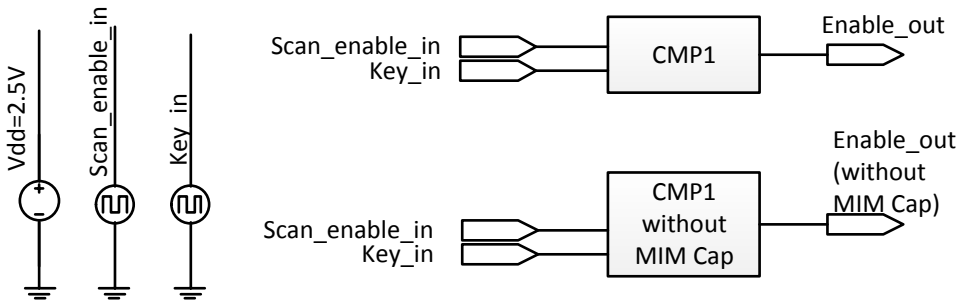


(b) Simulation of CMP0. The `scan_enable` signal is rising and `key_in` is fixed. Delta between M2 and M1 is the delay from input to output. `Enable_out` and `enable_out` without MIM switch at the same time. The MIM capacitor has no influence.

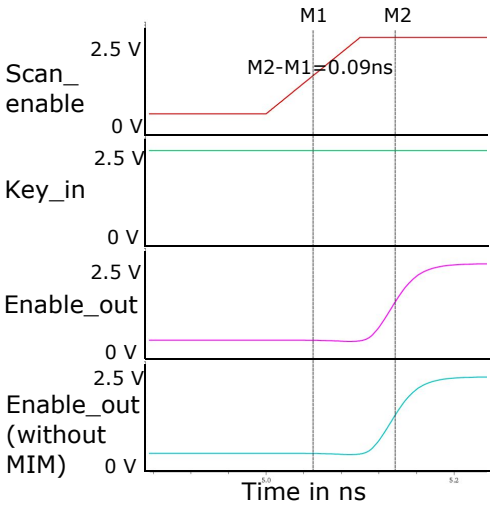


(c) Simulation of CMP0. The `scan_enable` signalis fix and `key_in` is falling. The MIM capacitor has minor influence. The markers M2 and M3 shows the additional delay and it is 0.03 ns.

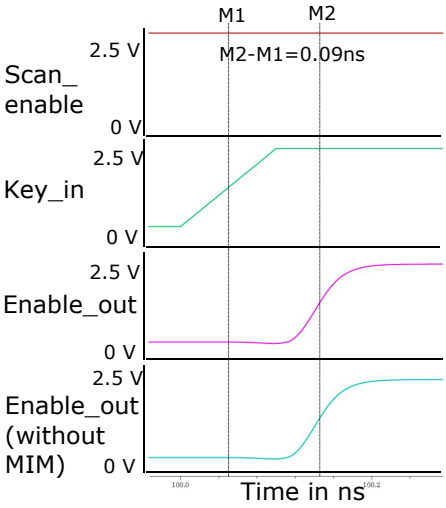
Figure 56: Simulation setup and results for CMP0. The compare unit is simulated with and without MIM capacitor to analyze the influence of the circuit. An additional circuit delay caused of the MIM structure is only if the `key_in` is rising of CMP0.



(a) Schematic for evaluation of CMP0. The security mechanism bases on a MIM capacitor, so the influence of the capacitor is simulated in this setup. The CMP0 unit is simulated twice. Once with the MIM capacitor, once without the MIM capacitor.



(b) Simulation of CMP0. The `scan_enable` signalis fix and `key_in` is falling. The MIM capacitor has minor influence. The markers M2 and M3 shows the additional delay and it is 0.03 Simulation of CMP1. The `scan_enable` signal is rising and `key_in` is fixed. The MIM capacitor has no influence.



(c) Simulation of CMP1. The `scan_enable` signal is fixed and `key_in` is rising. The MIM capacitor has no influence.

Figure 57: Simulation setup and results for CMP1. The compare unit is simulated with and without MIM capacitor to analyze the influence of the circuit. An additional circuit delay caused of the MIM structure is only if the `key_in` is rising of CMP0.

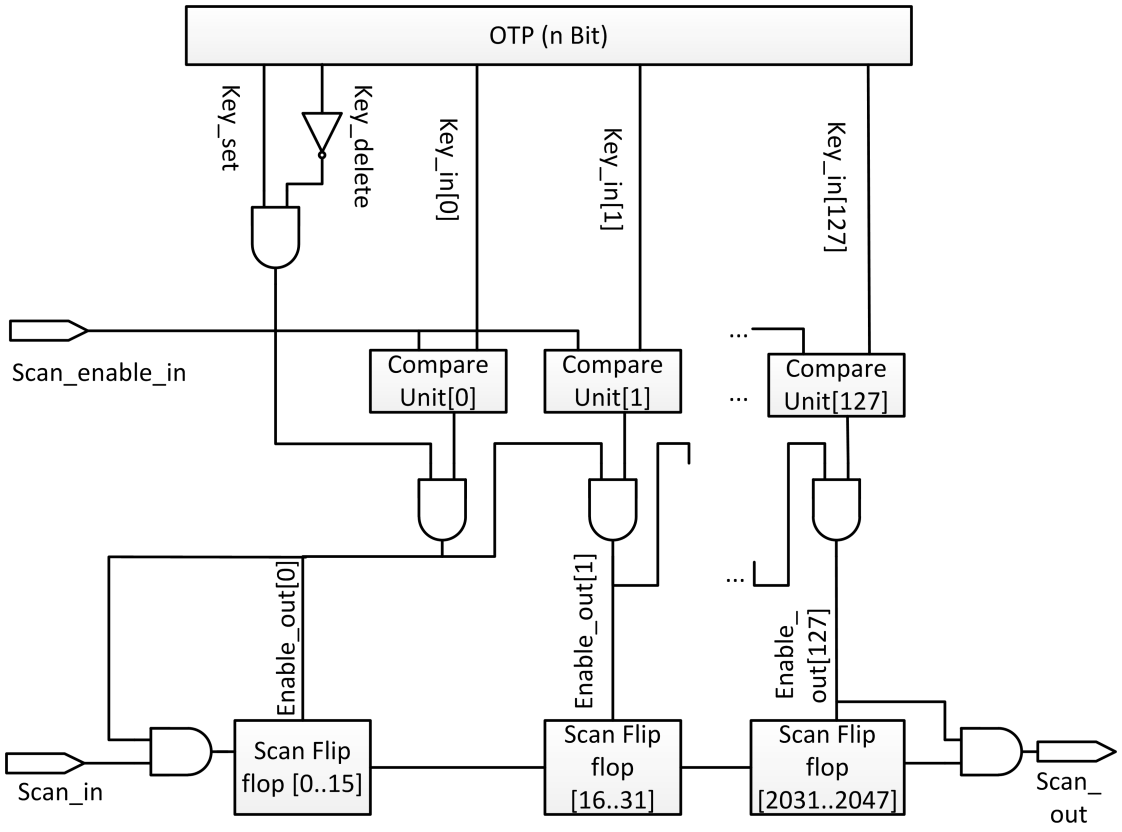


Figure 58: Block diagram of SSCI with AND module. Each compare unit follows an AND gate and each AND gate is the `scan_enable` input for a group of flip flops. This mechanism enables the scan chain only if all compare units gets the correct corresponding bit of the secret key.

flip flops or Flash can be reprogrammed so a brute-force attack cannot be excluded and with a matching key pair the scan chain is enabled. Clearly with a 128 bit secret key, a brute-force attack is nearly impossible. For smaller designs it might be an option to reduce the additional area for the secure scan chain interface and to reduce the length of the secret key.

For the slow scan chain interface there are no requirements for the timing. The additional area is legitimate as the ASIC is more secure than before. During test the DUT has no restricted power budget because this is done at the test environment which is hosted in a laboratory or fab.

After the key was deleted it should be not possible to rewrite the key again and further access to the scan chain it not possible. Thus, for the SSCI a one-time programmable memory is well usable.

The available OTP in IHP 0.25 technology is a one-time writable memory in anti-fuse technology based on transistors. After production every bit has the value "0" and can be set to "1" with a write operation. Before a test on test equipment is performed, the secret key has to be written into the memory. After the test the key is deleted. For an OTP the write time is approx. $100\,\mu s$.

In order to realize the secure scan chain the OTP array needs to be connected to the compare logic cells, and it needs to be programmed appropriately. If and only if all OTP cells are providing the values expected by the compare logic, the `scan_enable` signal is propagated to all scan flip flops.

As seen in Figure 58 every bit of the OTP and the corresponding compare cell is responsible to enable the scan functionality of more than one flip flop. E.g. if the scan chain has a length of $2{,}048\,\text{bit}$ and the width of the OTP is 128 bit, 16 flip flops share one security element. One security element for one flip flop would increase the security but the additional overhead is not justifiable. To enhance the security it is useful to avoid a straight forward mapping of the entries in OTP to the group of scan flip flops. It means that the first entry of the OTP should not enable the first 16 flip flops in the scan chain, the second entry the next 16 flip flops and so on. Otherwise a bitwise fixing of the secret key in the OTP might be easier. In fact a random assignment of a secure element to a group of scan flip flops seems to be a small security enhancement.

Design and Interface Description An OTP with a common interface for read and write is widely-used, e.g. for program memory in microcontrollers. A property of the used OTP in this thesis is that the memory has two different interfaces. A typical application area is a unique ID, like a serial number or a private key for an encryption scheme.

The OTP has two independent interfaces as shown in block diagram in Figure 59 and in layout in Figure 60. The first one is the programming interface for the read and write operation with external access. The OTP module as part of the secure scan chain solution is implemented in such a way that it is only writable at the programming interface and the `data_out` bus is not connected to the security controller as shown in Figure 59. The second interface is read only for the secret key and it is a parallel output for all stored data. The strict separation of both interfaces, one external for the programming operation and one internal for the read operation, secures the design against implementation errors which could cause a vulnerable device.

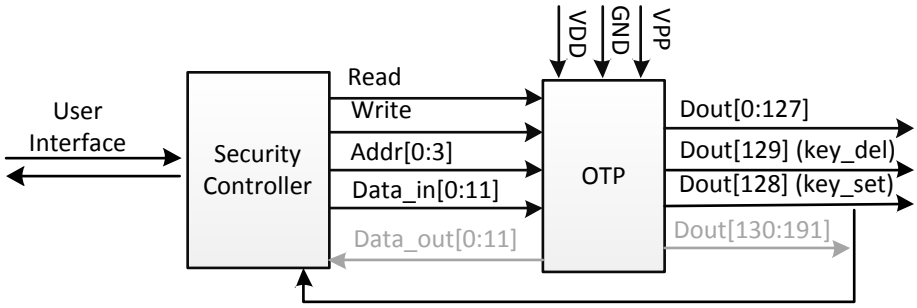


Figure 59: OTP unit for key storage, connected to the security controller. The `data_out` signal, marked in gray, is not connected to the security controller to avoid a read-out of the secret key via programming interface. The `dout[130:191]` is not connected because it is not used.

Since the programming interface has a 12 bit data bus and a 4 bit address bus to store 192 bit, the full parallel output does not need any address input. All bits are forwarded to the interface without any control signals and the content of the memory is permanently available at the output.

The used OTP has a size of 192 bit. This is related to the memory generator. It is possible to generate an OTP with a size of 128 bit, but at least two additional control bits are required, so that the next available size was chosen and some bits are not used.

Writing the Key The test engineer sets the secret key by using of the user interface of the SSCI. The security controller is responsible to perform this operation. At the serial interface a command is sent to the controller and it is decoded into a specific address and data word. Address, data and control signals are set and the write process is observed by the controller, too.

To enhance the security, a `key_set` bit is introduced. It is located in the OTP. As long as the `key_set` bit has the logical value “0” the secret key can be set, but the check of the key in the security elements, and the scan chain, is disabled. The value of the `key_set` bit can be changed only one time from “0” to “1”. With `key_set` is set to “1” the scan chain interface is enabled subject to the condition that the secret key is equal to the golden key.

Without this security feature a weak point exists. The assumption is that an attacker gets access to fully untested and non-programmed devices. The weak point is the following. A key contains “0” and “1” and the nature of the OTP is that per default the bit entry is “0” and it can be set to “1” in an irreversible step. An attacker would be able to set one bit and test the key. If the key does not match the attacker would set a further bit to “1”. This can be done until all bits are set to “1”. So an attacker is able to test up to 128 secret keys with one device. But with the additional security bit only one key can be tested. Clearly the disadvantage of this solution is that the setting of the key for a scan chain test by the test engineer has to be first time right. But this fact is negligible, since it is expected that the test setup is a well-structured process and mistakes are avoidable.

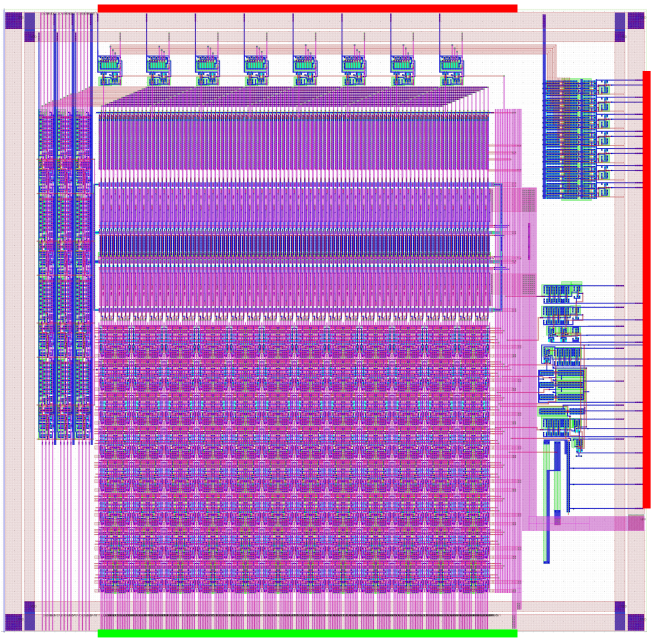


Figure 60: Hard macro of the OTP in IHP $0.25\,\mu\mathrm{m}$ technology. The unit is surrounded by a power ring and the size is $305.7\,\mu\mathrm{m}$ in the height and $315.8\,\mu\mathrm{m}$ in the width. The programming interface is located at the top and at the right side, marked with red lines, and the parallel output `dout` at the bottom of the macro, marked with a green line.

Deletion of the Key The only operation which can be performed with the stored secret key is to delete the secret key. Thus as soon as the test is completed, successful or not, the scan chain has to be deactivated, i.e. all the OTP bits have to be programmed, which means all bits are set to "1". To minimize weak points the user cannot delete the key bit by bit or word by word. The user can only send a command per user interface to the controller, which causes the deletion of the secret key. This avoids that the user forgets to delete parts of the secret key or to delete only parts of the key in bad faith. Furthermore the status flag `key_del` in the OTP is set automatically if the complete secret key is deleted.

Power Supply A requirement of OTP memory are the two different voltage levels for the power supply, depending of the operation mode. To change the value of a cell from "0" to "1" a high voltage at the gate of the fuse transistor is required. This high voltage has to be applied only for programming which exceeds the breakdown voltage and it is typically above $7\,\mathrm{V}$. The read operation is performed at the nominal operation voltage, e.g., of 2.5 V for the used $0.25\,\mu\mathrm{m}$ IHP technology.

To provide the programming voltage two different ways are possible. The first one is to use an external voltage source which requires an additional pad. To built-in an additional pad is the most suitable solution with low effort for designers. Only to set and delete the secret key the high voltage has to be applied to the device at this pad. In read phase a voltage of $2.5\,\mathrm{V}$ is required at the input of the pad. If the output data of the OTP is not necessary, the power pad can be connected to ground.

The second approach is an internal circuit which generates the required voltage. A similar step-up converter for flash memory has a size of $450\,\mu\mathrm{m} \times 180\,\mu\mathrm{m}$ and special high voltage transistors are necessary. Furthermore high effort for schematic and layout generation is required.

Power Saving Strategy In the area of wireless sensor nodes one of the requirements is a low leakage and a low operation current of the device in field. Any unused component should be in a power saving or power-off mode. Different power saving strategies are explained in [31]. Due to the fact that the OTP is a block design with its own power rings, the power of the unit can be switched off after a successful test. In [59] power gating cells are described, which are well usable for this application area. In normal operation mode the OTP consumes no power anymore.

Partitioning With a size of 192 bit the OTP has more entries than the 128 bit key requires. Two additional bits are used for further security mechanisms. The status flags "key_set" and "key_del" are readable by the user over the programming interface. All other entries of the OTP which are related to the secret key and the secure scan chain interface are not forwarded for reading.

Since some bits of the OTP are not used, it can be used to store other values like the device ID and so on. But from security point of view, it is not recommended to use the OTP for additional function. On one hand, the power saving strategy cannot be implemented; on the other hand any design modification can cause a security leakage.

4.4.4 User Defined Interface

To perform the scan chain test, the secret key has to be written into the device. In this thesis an SPI was used for the example implementation. The interface component and all other components of the wrapper are not part of the scan chain; they are used before the scan chain test is done. The test of the interface component is done in the functional test during setting the secret key. If this action fails, the ASIC is marked as "failed device".

4.4.5 SSCI Controller

The controller connects compare units, external programming interface (SPI) and OTP. Furthermore some functions like set and delete of the key are implemented to minimize handling errors and security leakages. Controller and SPI are connected to the system clock. The slight logic deep allows a high operation frequency.

The used SPI as communication interface is a serial protocol and the controller contains an SPI slave. The signals `clock`, `CS`, `MOSI` and `MISO` are part of the interface and they are connected to the test equipment. In Table 7 an overview of supported commands is given. At the external programming interface a command identifier, followed by data and address are set. The SPI module has a 32 bit internal output. Several bits are used to select the command, 12 bits are mapped to data input and further 4 bits are mapped to address input of the OTP.

The used OTP has a memory interface and a data width of 12 bit, which is enforced by the memory generator. So, for the 128 bit secret key 11 write transactions with 12 bits are necessary. After the procedure of writing the key, a `key_set` bit has to be set to enable the scan chain functionality. In sum 12 write transaction, each 32 bit, are necessary. It results into 384 clock cycles at the SPI port.

In case that the user writes the "key delete" command, all data inputs at all addresses are programmed with a "1" automatically by the controller. At the end of the procedure, the key is deleted and the scan chain cannot be used anymore.

Table 7: Control commands of SSCI

Function	*Command (32 bit)*
Write Key	0x0100 + data<11:0> + addr<3:0>
Set key active	0x020...0
Delete key	0x030...0

4.4.6 Verification

The verification consists of two different steps. The first step is the verification of the design concept. It means that the concept of SSCI has to be reviewed against design errors. The second step is the verification of the implementation, which means that the the system has to be simulated extensively with a high test coverage. The implementation is simulated as stand-alone module with its own test bench as shown in Figure 61.

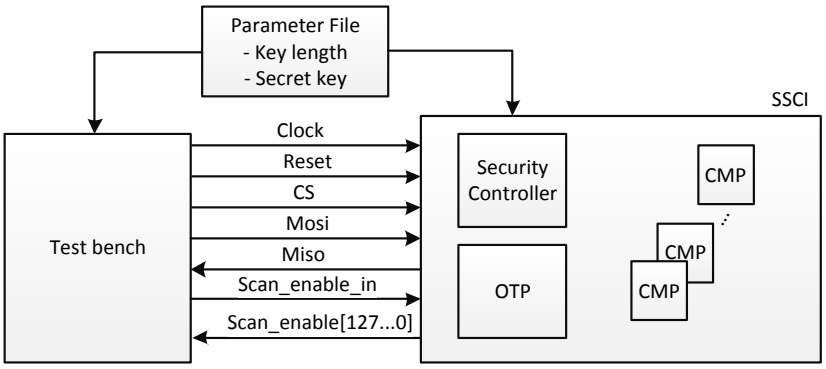


Figure 61: Block diagram of SSCI and test bench

4.5 Integration and Adaption of the Target Design

From a design in HDL to a shipped device, many steps are necessary. An additional component can have effects at different steps in the design flow. In the following the integration of the SSCI into core is described and in every step the impact of the SSCI is checked.

4.5.1 Technological Prerequisites

The first requirement to implement the SSCI is that the technology supports a thin isolator layer between two metallization layers, which is typically used for a MIM capacitor. To get a high capacitance such an isolation layer is very thin, and during optical inspection of a circuit, invisible. On the element of the MIM capacitor the security feature of this thesis was built.

Furthermore for applying the solution sketched above, it is essential that the used technology offers OTPs to store the secret key. For this, especially the anti-fuse OTP is well suited, since the basic cell can be implemented in any CMOS technology and consists of three transistors only.

4.5.2 Influence on Design Flow

In Figure 62 a typical design flow, from RTL to shipment, is shown. The stages, marked in green color, are the additional steps which are required to integrate the SSCI. These steps are explained in detail in the following.

4.5.3 RTL Design

The SSCI is fully implemented in HDL and ready to use and all SSCI components are combined in one module, which simplifies the integration. The only change of the source code before the component can be used, is related to the secret key. The default secret key into the parameter file has to be replaced by a new one. Target system is developed like every other processor without any consideration that a secure scan chain interface is inserted in a later phase in the design flow.

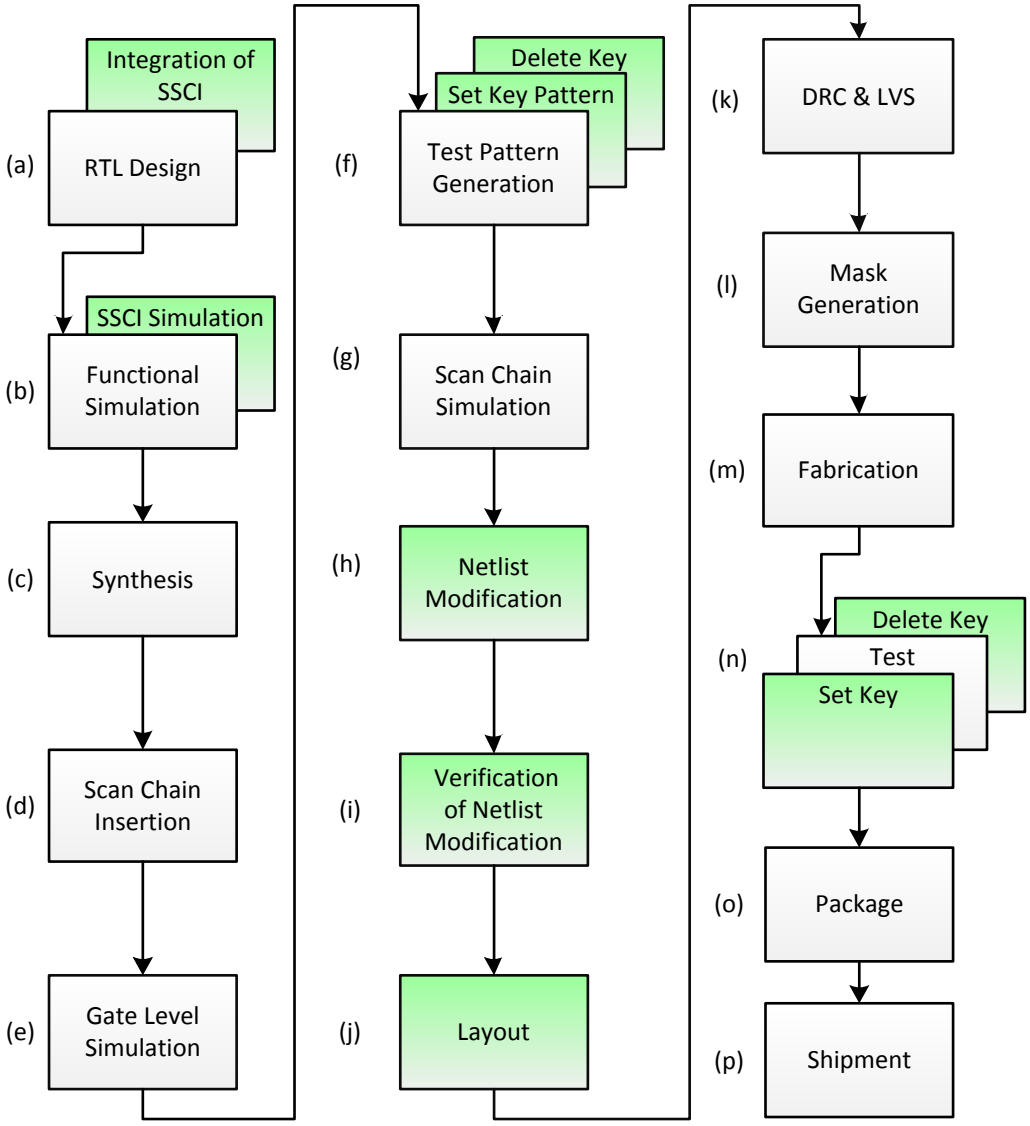


Figure 62: Detailed design flow for ASIC development and manufacturing. Steps with additional effort to integrate the SSCI are marked in green.

Listing 2: ifdef in verilog

```
1 `ifdef ASIC
2   //implement full SSCI here including compare units and OTP
3 `elsif FPGA
4   //implement SSCI here.
5   //Replace OTP by RAM and do not implement compare units
6 `else
7   //nothing
8 `endif
```

In the next step the system core and the SSCI are merged into one design and the core which has to be protected is connected to the SSCI module.

4.5.4 Simulation

Since a stand-alone test bench for the SSCI is provided in this work, the test bench of the target is device specific. An error during the integration process could cause an insecure scan chain interface, an untestable device or even a fully non-working device. Keep in mind that two different simulation setups are required. At first the system has to be simulated to ensure that the new component does not influence the functionality. This is equal to a system test. After this the functionality of the integrated SSCI has to be simulated.

In step (b) in Figure 62 only a functional test of the interface and the key set and key delete is performed. The scan chain test cannot be simulated in this phase because it is not integrated yet. For the simulation in the digital design flow the compare units are part of the netlist and base on digital standard cells. The OTP is based on physical destroying of a gate of a transistor. This is a physical effect, which cannot be implemented in HDL. As workaround a behavioral model for the simulation is implemented. The function is to emulate writing the secret key as well as deleting the secret key.

Often after a simulation on RTL a design is verified on an FPGA. For pure digital designs this is a fast possibility for verification. The OTP and the compare cells cannot be mapped to FPGA elements directly. But the implementation of these modules in an FPGA is not necessary, because they are used in case of a scan chain test only. It is useful to test the programming interface of the SSCI on an FPGA. To enable such a test, in HDL code macro directives are used. In a preprocessing step the corresponding code section is selected. An example is given in Listing 2. So for an ASIC the SSCI is fully implemented as required, for the verification on FPGA the SSCI consists of the programming interface and a small memory block only. The memory, e.g. an SRAM, is the substitute for the OTP, so that the functionality of the interface can be verified.

Apart from this small modification in the SSCI module, the design can be implemented for FPGA verification as before. Clearly the scan chain cannot be tested, but the test on the FPGA is for a functional verification of the design and not for verification of the test approach.

4.5.5 Synthesis and Scan Chain Insertion

The following steps, (c) and (d) in Figure 62, synthesis and scan chain insertion, are done without any changes in the design flow. The HDL code is transformed into a gate level netlist and a scan chain is inserted.

The controller of the SSCI is written in HDL and it is synthesizable using DesignCompiler [75] like any other component on RTL. The compare units are combined in a netlist already and the OTP is generated as a hard macro. A difference between the protected core and the SSCI is that the SSCI does not get a scan chain.

4.5.6 Test Pattern Generation

Usually a test bench is generated by the scan pattern generator. For this tool the protection mechanism is a transparent solution.

During test pattern generation the additional test patterns for set and delete key have to be generated. They are used later during post-production test before shipment of the device.

4.5.7 Scan Chain Simulation

After scan insertion a simulation of the scan patterns has to be done. In Figure 63 the procedure of this step is shown. At first the scan chain interface has to be enabled using the correct secret key, afterwards the scan chain test is performed and at the end the secret key has to be deleted. Clearly the deletion of the key is not a necessary step to simulate the scan chain test, but it is useful to verify this feature. The effort for the design team to develop the test bench to write and delete the secret key is minor. The basis for these test benches are the test benches for the stand-alone verification of the SSCI. Based on this test bench the patterns for the test equipment are generated.

4.5.8 Netlist Modification and Verification

After synthesis and scan insertion the netlist of the core has to be slightly modified in step (h) in Figure 62. The netlist has a single input for the `scan_enable` signal after scan insertion and all flip flops are switching from functional mode to scan mode by enabling this signal. To avoid this weak point the number of `scan_enable` signals has to be identical to the number of compare cells. Instead of a single `scan_enable` signal for all flip flops, a group of flip flops get their own `scan_enable` signal. In this step the single signal is replaced by a bus for the `scan_enable` signal. E.g. in a design with 2,048 flip flops and 128 compare cells, the `scan_enable` bus is 128 wide and each group of 16 flip flops has its own control signal to switch in scan mode.

A TCL script is provided in this work to perform a modification of the netlist. The user sets the number of key bits in the configuration section of the script. The `scan_enable` signal at the input of each scan flip flop is replaced by a selected wire from the `scan_enable` bus as shown in Figure 64 during the modification step. The replacement can be done for a single scan chain as well as multiple scan chains in a design.

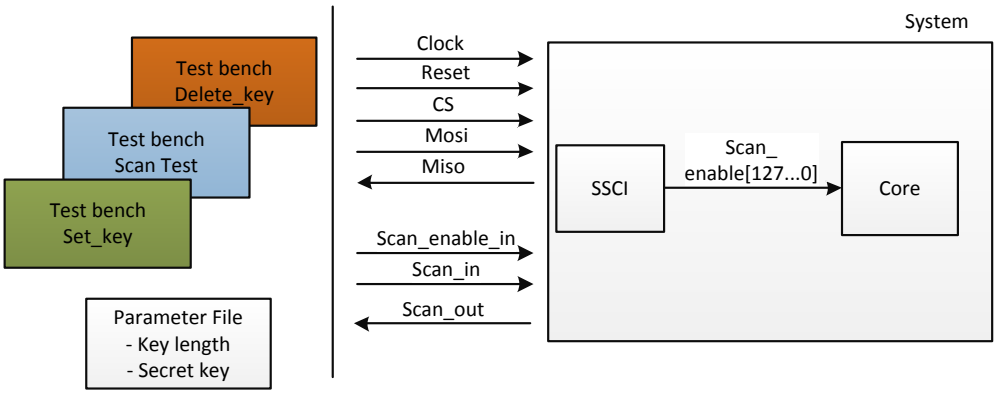


Figure 63: Simulation of the system with integrated SSCI for verification of the scan chain. The first test bench (marked in green) sets the secret key to enable the scan functionality. The second test bench (marked in blue) is the generated test from the test pattern generator and it performs the scan chain test itself. The third test bench (marked in orange) disables the scan functionality and verifies the deletion process of the secret key.

This slight modification of the netlist is fail-safe, but a verification step, for example with a netlist equivalence checker, is clearly required (step (i) in Figure 62). In Formality (Synopsys) [77] the original and the modified netlist are read in. A compare rule defines a mapping of the `scan_enable` bus in the modified netlist to the single `scan_enable` in the original netlist. Afterwards the tool verifies the equivalence of both netlists. This kind of verification avoids an extensive simulation of the modified netlist and it is fast and safe.

4.5.9 Layout

In the layout phase of a system with integrated SSCI the “don’t touch” attribute has to be set for the SSCI. This is required to avoid that the layout tool performs any optimization which could influence the functionality. The control logic and the compare units are standard cells and they are distributed on power rails besides all other standard cells of the processor. The OTP is a hard macro cell, which means that it is a fully pre-layouted design with pins for inputs, outputs and power supply. In case power gating is used to allow a deep sleep mode of this memory, special power gating cells have to be inserted.

To optimize the timing and area utilization the layout tools offer a command to reorder the scan chain. After synthesis the order of the scan chain, this means the output of one flip flop to the input of another flip flop, is more or less random. It could cause that a flip flop in one corner of the ASIC is connected to a flip flop in the opposite one. The effort for the wiring is increased. The process of reordering the scan chain reduces this effort to a minimum, because each flip flop in the scan chain is connected to its nearest neighbor. The functionality of the design is not

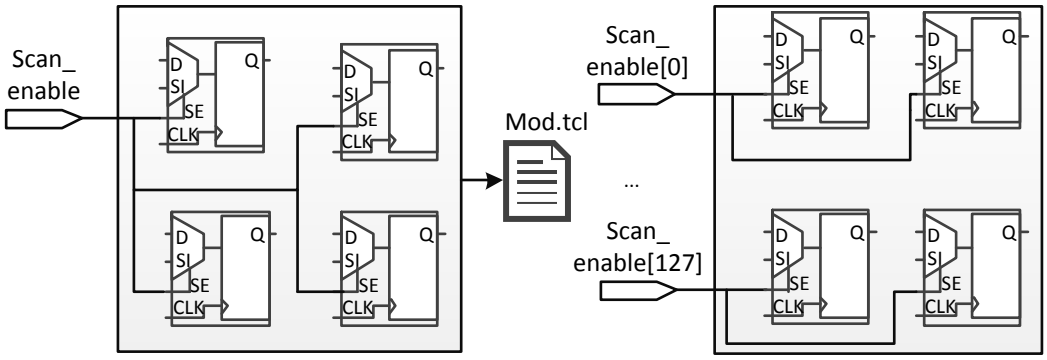


Figure 64: Netlist modification step to replace single bit `scan_enable` by 128 bit bus `scan_enable`

influenced by this modification. Only the test vectors have to be generated again. For timing and area critical designs a reordering of the scan chain is indispensable, but solutions like proposed in [67] and [66] and many other scan chain protection mechanism do not support it.

4.5.10 Test

During the test phase additional test pattern have to applied to the DUT as described in Section 4.6 in detail.

4.6 Test

Goal of the developed SSCI is at the one hand an effective security mechanism and on the other hand to ensure that the design is fully testable without any restrictions. In the following section the process of test for a design with SSCI is described.

The OTP, which is built-in as storage for the secret key, requires a high programing voltage. The discussion in Section 4.4.3 concludes that this voltage is applied on a dedicated input pad. For this case the test environment has to deliver the programming voltage. To provide different voltages, e.g. for core and pad power supply, and even for analogue circuits, is a standard function of test equipment. So the voltage for the OTP can be delivered without any additional means.

4.6.1 Process of Scan Pattern Generation

For scan tests a set of test pattern is necessary. These patterns are generated by tools like Tetra-Max (Synopsys) [78]. In a DRC the test pattern tool verifies that all scan flip flops are fully

controllable. To control a scan flip flop in test mode all inputs and outputs of the flip flop have to be accessible via the scan interface. Thus the clock signal, data input and data output as well as the `scan_enable` signal have to be controllable in the scan mode.

Two different cases have to be considered for pattern generation. The first case is the generation of pattern of the original netlist directly after synthesis. This can be done as for any other scan design. Any flip flop is controllable with `scan_enable` signal and the design passed the DRC.

The second case has to be considered, if the scan chain was reordered in the layout phase as described in Section 4.5.2. The netlist contains the compare units and they influence the `scan_enable` signal. Therefor the secret key has to be applied to the input of the compare units using corresponding commands in the test pattern generator. With the correct secret key, applied at the input, the `scan_enable` signal is controllable and DRC is passed.

After this, the generation of the test pattern is the same as for any other case. Also any type of test pattern can be generated, e.g. test on stack-at faults or even a path delay test. The number of test vectors is equal to a design without SSCI.

4.6.2 Test Process

In contrast to an unprotected design, two additional patterns are required to perform the test: the first pattern for setting the secret key before the test and the second pattern to delete the secret key after the test. In Figure 65 the step for setting the secret key is marked in green and the step for deleting it in orange. These patterns can be generated in a simulation as described in Section 4.5.7 and they are in VCD or in EVCD format. The test patterns from TetraMax are in WGL or in STIL format. During the setup phase the test engineer integrates all test patterns, even in different formats, into the test program.

For writing the key a certain time interval is required. At first the sending the secret key into the device at the SPI requires 384 clock cycles. Second the writing of the secret key into the OTP. To write one data word into this memory a time of $100\,\mu s$ and for the full secret key a time of $1.2\,ms$ is required. If the device has a system clock of 10 MHz, additional $12{,}384$ clock cycles are necessary for the setup. After the post-production test, the secret key has to be deleted. The deletion of the secret key is equal to a write process and all bits are written. Again a time of $1.2\,ms$ is necessary to deactivate the SSCI.

The handling of the test pattern file, which contains more precisely the secret key, has to be done in a secure way. A file transfer should be done using Secure Copy (SCP) or FTP over Secure Socket Layer (SSL). Also the access rights of the data should be restricted to trusted employees only. Nevertheless the secret key in hands of an attacker is not sufficient for a successful attack. The attacker needs an untested ASIC to enable the secure scan chain in a fast way.

4.6.3 Support of Several Test Cycles

In times of 3D integration, each die can be tested separately and after stacked bond procedure an additional test can be performed again. There are two different possibilities to solve the problem

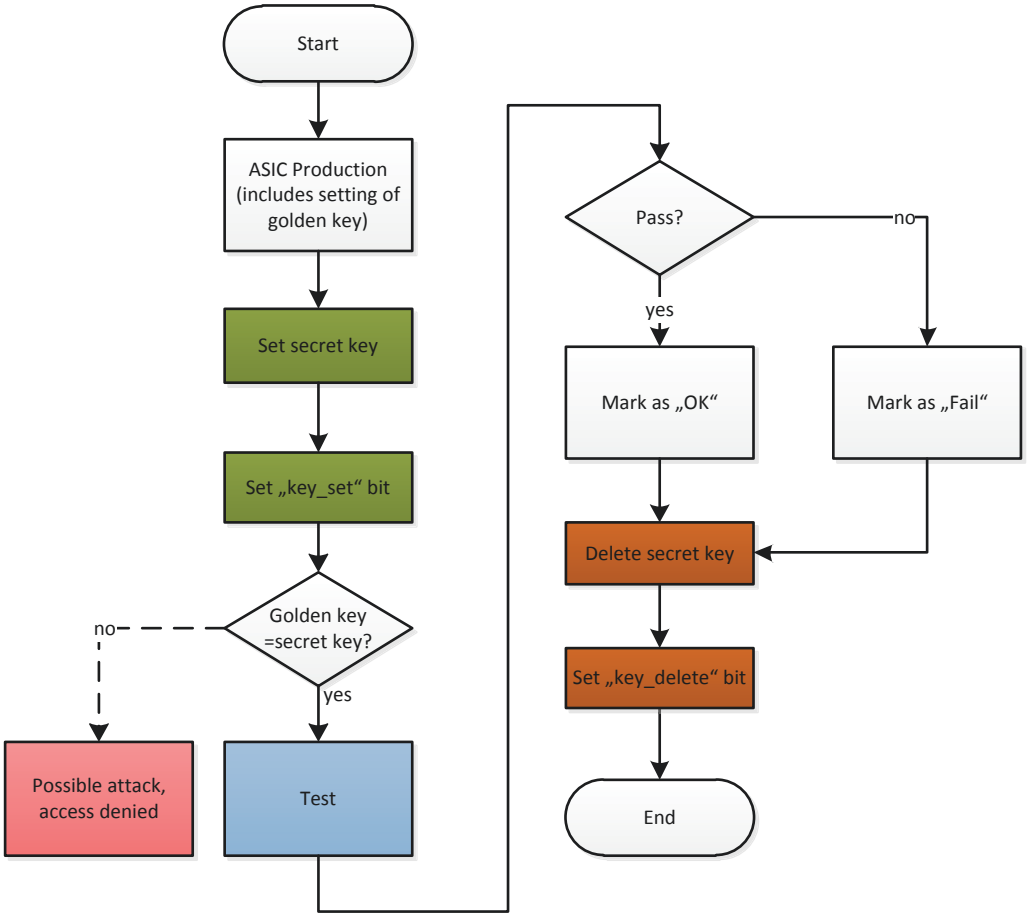


Figure 65: Test flow for an ASIC with SSCI. The steps for setting the secret key, marked in green, and the steps for deleting the secret key, marked in orange, are additional to the standard test flow.

of an additional test. The first one is not to delete secret key after test. The secret key is deleted after the final integration test, only.

The second possibility to perform a second test is a second row in the OTP. After the first test of the die, the secret key is deleted. In the final test, after the 3D integration, the secret key is written into the second row of the OTP and the scan chain interface is re-enabled until this key is deleted too.

The same approach, i.e. a second row in the OTP can be used for an aging test. In the first post production test the first entry in the OTP is used. After a certain time in operation the device is tested again. In this test the second row in OTP is used. Since the scan chain test can be done without any limitations even an error analysis can be done. This is helpful to identify types of digital standard cells, which can be improved to minimize aging effects.

4.6.4 Advantages of the Proposed Solution

The generated test patterns from the scan pattern tool can be used without any modification. This is an important fact, because the test pattern generators are not adaptable with respect to secure scan chain interfaces. Furthermore any compression technique for test patterns can be used combined with the proposed SSCI.

Even a test, in which the maximum operation frequency of the device is determined, can be performed. This test requires one test pattern, followed by two operation clock cycles and a read out of the data via the scan chain. This kind of test is important to classify the produced devices into different speed grades. Once the scan interface is enabled, it is not influenced by the protection mechanism.

4.6.5 Potential Weaknesses

The secret key is written into the device without an authentication scheme. Without this security mechanism, everyone is able to set the key on a certain value and to delete the key. The disadvantage of an authentication scheme is the required area for a hardware implementation of the ECC algorithm as described in [18]. Beside that everyone is able to set the secret key in the device; every device is able to receive the secret key. E.g. an attacker can grab the secret key during a test with a special prepared ASIC. This ASIC needs an SPI to receive the commands with the secret key, coming from the test equipment. An NVM stores the data permanently. In a later step the attacker reads out the memory and extracts the secret key.

A further weak point is the physical access during the test procedure. If the DUT is not protected against physical access, an attacker can remove a device during a running test. As result he gets a device in which the scan interface is still enabled.

An access control for employees in the field of ASIC production and test is self-evident and the trusted environment minimizes the problems above.

4.7 Evaluation

In this section the SSCI is analyzed with respect to its influence on the target device.

4.7.1 Influence on the Target

The main influence on the target system is applied to the scan chain interface. An additional component is integrated to support the feature to unlock and lock the SSCI. This means for the test case the system gets an additional function: write and delete of the secret key. All other components are untouched. After a test and deleting the secret key it is no longer possible to enable the shift register functionality of a scan chain.

4.7.2 Area

In Table 8 the area of the compare units and the AND gates is given. The AND gates are responsible for ensuring that the output of the `scan_enable` bus is only activated, if all compare

Table 8: Area of compare units (128 bit) including AND module.

Cell type	Number of Instances	Area
CMP	128	$7{,}224\,\mu m^2$
AND gates [1]	128	$5{,}418\,\mu m^2$
Sum		$12{,}642\,\mu m^2$

[1] 128 AND gates are the AND module

Table 9: Area of SSCI after synthesis and estimated area for layout.

Component	Area (synthesis)	Area (layout)[1]
Controller	$0.003\,mm^2$	$0.004\,mm^2$
SPI slave	$0.011\,mm^2$	$0.014\,mm^2$
Compare cells	$0.013\,mm^2$	$0.017\,mm^2$
OTP	- [2]	$0.097\,mm^2$
Sum		$0.132\,mm^2$

[1] Estimated
[2] Hard macro, layout only

units have a valid output. These compare units and AND gates are special modules and fix in structure and area for a given key length.

The SSCI controller and the SPI component are written in HDL and the required area differs, depending of the synthesis tool. The synthesis of both components was done by Synopsys DC [75]. The OTP is generated with a memory generator and integrated as hard macro. In Table 9 an overview about the required area for all components is given.

For controller, interface and compare cells the additional area overhead for routing is not considered in the synthesis phase. Typically $30\,\%$ of the area is added during placement. This value is added in the second column of Table 9 to get more accurate area estimation.

The netlist after synthesis has to be modified as described in Section 4.5. In this process the `scan_enable` input is split to a bus. Since only the wiring is changed, the input load of all cells is identical to one of the non-modified netlist. Additional buffers are not required and the area is not influenced.

The additional area required for the SSCI is relatively small compared to the area of a full processor. For example the core of the openMSP430 requires an area of $15.8\,mm^2$ in the layout phase including cryptographic cores and memories. In Table 9 the area after layout for the SSCI is given with $0.132\,mm^2$. This means that area overhead for the secure scan chain interface is less than $1\,\%$ of the protected system. For benchmarking issues the area is equal to 5,196 GE [2].

[2] A gate equivalent is $25.4\,\mu m^2$ in $0.25\,\mu m$ IHP Technology

4.7.3 Power

In the following the power consumption for different cases is described. The analysis of the power consumption as well as the one of the timing was done with a typical nominal voltage of $2.5\,\mathrm{V}$ and a temperature of $25\,^{\circ}\mathrm{C}$ for the $0.25\,\mu\mathrm{m}$ IHP technology.

Power in Test Mode In the post-production test, the power is provided by the power supply of the test equipment. During a scan chain test a high number of gates switches at the same time which causes a high current flow. It is important that the additional security mechanism for the scan chain does not increase the power consumption significantly. Otherwise the ASIC is at risk, due to a damaged power supply pad because a high current in pad and the temperature is increased to a dangerous level.

In test mode the additional SSCI component does not influence the power consumption of the system significantly. The `scan_enable` signal is a simple input. Due to the fact that high number of a flip flops is driven by the `scan_enable` signal, a buffer tree is required. After integration of the SSCI, a similar buffer tree is required because the input capacitance of all connected flip flops is the same as without SSCI.

For the compare units only combinatorial cells are used. This means that the units consumes power only if the input is changed. The input is either the secret key from the OTP or the `scan_enable` signal. The secret key from the OTP is fixed most of the time and is only changed to enable and to disable the scan chain interface. In contrast to the key the `scan_enable` signal is changed during a scan test for each test vector.

To assert the `scan_enable` signal, the compare units require an energy of $8.88\,\mathrm{pW\,s}$ for the example design with a 128 bit secret key. To disable the shift register functionality, which means setting the `scan_enable` signal to low, consumes $9.72\,\mathrm{pW\,s}$. In sum an energy of $18.6\,\mathrm{pW\,s}$ is consumed by the compare units for one scan test vector. The openMSP430 requires an energy of $2.14\,\mu\mathrm{W\,s}$ to shift in one scan test vector. This means that the additional power overhead is negligible and the power consumption during the scan chain test is not influenced significant.

Power in Operation Mode The power in the operation mode has to be considered for two different cases. The first case is the operation cycle during scan chain test and the second case is the operation mode when the scan chain is fully disabled. In both cases the input of the compare units is static. The `scan_enable` signal is low and all bits of the secret key from the OTP are at a defined value. So the dynamic power consumption is zero.

Leakage For sensor nodes the standby current is an important factor to get a long battery life time. Most of the time the device is in standby mode and the leakage dominates the power consumption. The power consumption of the controller and the OTP can be reduced by using a power gating technology. Controller and OTP are separate modules with their own power supply rings. In these rings the power gates are integrated and the components can be switched off to save energy. Without power gating the controller has a leakage of $11.6\,\mathrm{nW}$. For the OTP the leakage is not given in the data sheet, so this value cannot be considered.

For the compare units the approach of power gating is not useful, because the digital standard cells are distributed in the core area. Only the leakage power of the compare units and the AND gates increases the power consumption in operation mode.

In Table 10 the different cases for the leakage analysis are described. Every digital standard cell has a leakage current, which depends of the cell type and the value of the inputs. In column 3 and 4 the leakage is estimated for the case that a secret key is stored in the OTP. In both cases, `scan_enable` signal on and off, have to be considered. The values for the leakage in operation mode are given in column 5 of Table 10. The cells are not in a separate power island and cannot be disconnected from the power supply. The leakage is approximately $10.5\,\mathrm{nW}$. The openMSP430 has a leakage of $4.3\,\mu\mathrm{W}$, which means that the additional cells for the SSCI increase the leakage of the protected design by $0.2\,\%$ only.

Table 10: Leakage of compare units and AND gates. These cells are distributed in the layout and cannot be disconnected from the power supply in non-test mode. For the compare units a set of 64 CMP0 and 64 CMP1 is assumed.

Cell	*#*	*Secret key set, `scan_enable='1'`*	*Secret key set, `scan_enable='0'`*	*Secret key deleted, interface disabled*
Compare unit	128	7,210 pW	5,420 pW	5,890 pW
AND gate	128	5,403 pW	4,587 pW	4,587 pW
Sum	256	12,613 pW	10,007 pW	10,477 pW

4.7.4 Timing

The developed approach for protecting the scan chain influences the timing in the test mode. The design itself is not changed and the protection mechanism influences the `scan_enable` signal only. The `scan_enable` signal is an input of the compare units, which have an internal delay.

In Table 11 and 12 the path delay for each type is shown. Both types of compare units have the same delay of $0.09\,\mathrm{ns}$. In Figure 66 and 67 the schematic for both types of compare cells is shown, which helps to understand why both cell types have the same internal delay. The data value of the `key_in` signal is fixed as long as the OTP has the same content. At the beginning of a scan chain test, the secret key is written into the device. From this point on, the key is fixed at the input of the compare cells, until the secret key is deleted after a successful test. Due to the static key input the inverter U1 has a static output for CMP0 and for CMP1 the inverter U1 is fully bypassed. The internal delay of the cell depends on changes of the `scan_enable_in` signal only.

For the example design with a 128 bit key, the enable signal has to be forwarded through 128 gates as shown in Figure 68. For the $0.25\,\mu\mathrm{m}$ IHP technology the `scan_enable_in` signal is propagated to all flip flops in approximately $10\,\mathrm{ns}$ if the secret key is correct. In Figure 69 the simulation for this case is shown. This additional delay has to be considered when the `scan_enable_in` signal is set. During a scan test this is required before a scan pattern is loaded into the DUT. The clock frequency which is used to shift data into the scan chain is

Table 11: Delay of Compare Unit CMP0 with fixed `key_in` signal and variable `scan_enable_in` signal

Cell type and name	*Delay*
Inverter U1	0.00 ns
NAND U2	0.06 ns
Inverter U3	0.03 ns
Sum	0.09 ns

Table 12: Delay of Compare Unit CMP1 with fixed `key_in` signal and variable `scan_enable_in` signal

Cell type and name	*Delay*
Inverter U1	0.00 ns
NAND U2	0.06 ns
Inverter U3	0.03 ns
Sum	0.09 ns

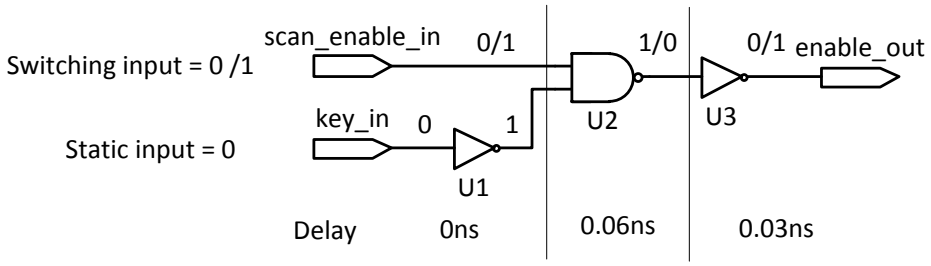


Figure 66: Schematic for timing analysis of CMP0

low; typically the period is 100 ns. Thus the additional delay does not influence the timing constraints for test patterns. The same internal delay of CMP0 and CMP1 prevents an extraction of the secret key. For a 128 bit secret key the delay to enable the scan chain is the same.

In case of setting the `scan_enable_in` signal to logic "0", the information is propagated in 0.27 ns to each scan flip flop as shown in the simulation in Figure 70. The delay in test mode is necessary and cannot be improved without influence into area and power consumption.

In summary, the most important fact is that the timing in the operation mode is not affected. Since the logic path is not changed, the additional circuit does not cause a delay. This is an important fact to prevent an attack, which observes the timing behavior.

4.7.5 Adaptability and Scalability

A well designed protection mechanism is well designed only if it can be reused for any system without circumstances or modifications. The main idea behind the SSCI, is to control the `scan_enable_in` signal. Since scan flip flops have a common architecture, it can be used for different technologies which uses traditional scan flip flops or even scan latches. A precondition is that the used technology supports MIM capacitors to contribute the required invisible isolator. Without this isolator the developed approach works as well, but the compare units for "0" and "1" have an individual foot print in layout. An attack based on reverse engineering would be easier.

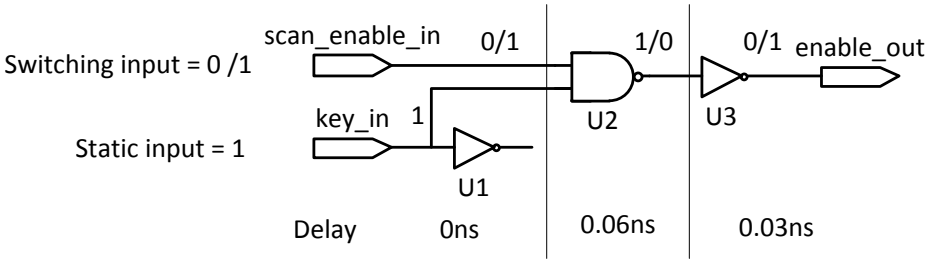


Figure 67: Schematic for timing analysis of CMP1

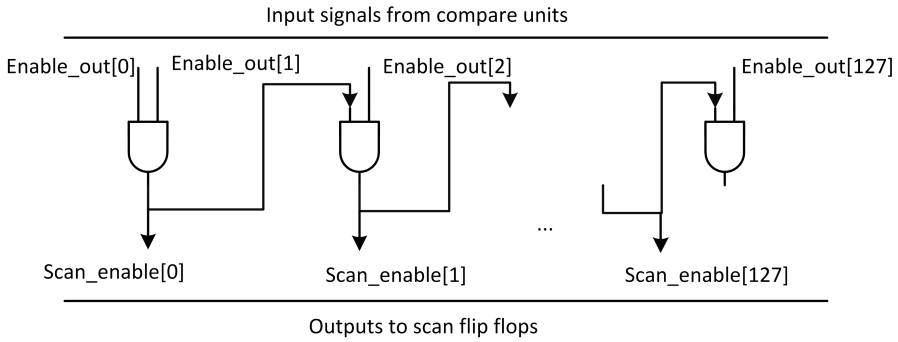


Figure 68: Logic path for AND module. The last scan_enable signal (`scan_enable[127]`) is activated, if the corresponding bit of the secret key is correct and all other secret key bits before are correct, too. This causes a long logic path and a long propagation time.

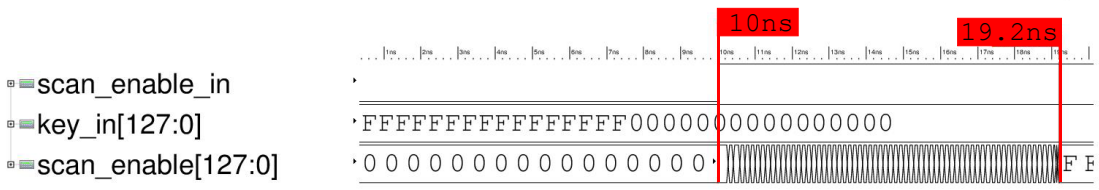


Figure 69: Delay for `scan_enable_in` signal. At 10 ns the scan chain functionality is switched on and at 19.2 ns the `scan_enable` bus is fully switched on. The propagation delay is 9.2 ns.

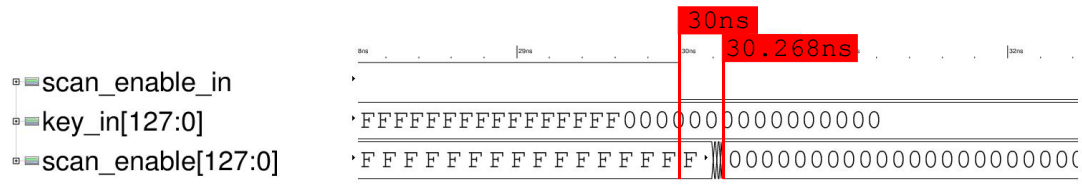


Figure 70: Delay for releasing the `scan_enable_in` signal. At 30 ns the scan chain functionality is switched off and at 30.27 ns the `scan_enable` bus is fully switched off. The propagation delay is 0.27 ns.

In the chosen example the secret key has a length of 128 bit. For every bit of the OTP one compare unit has to be integrated into the design. Any adaption of the length of the secret key is realizable with low effort. As described in Section 4.9.1 a generic verilog implementation is provided in this work. It is adaptable regarding key length and key value. Even the test bench for the stand alone simulation is automatically adapted.

Furthermore only the `scan_enable` signal is touched, independent whether the design has one scan chain or a lot of them. The area is not influenced by this fact. Even the number of scan flip flops is not restricted by this approach. The power consumption, area and timing are independent of the design size and depend only on the key length.

The minimum number of flip flops in a secured design is equal to the number of bits of the key. In this case for every flip flop one bit of the key is responsible. An upscaling of the number of scan flip flops to any value is possible. Independent of the number of flip flops, all bits of the secret key have to be set to the correct value to enable the scan chain.

The provided TCL script, for modifying the netlist, counts the number of scan flip flops. From the number of scan flip flops into the design and the number of key bits, the script calculates the number of scan flip flops with a shared `scan_enable` signal.

Vendors of soft IP blocks offer encrypted netlists to protect their knowledge. In this case it is necessary that the vendor executes the TCL script before netlist encryption. Afterwards the encrypted netlist can be integrated by the customer in its system like any other component.

4.8 Security Analysis

In the following selected attacks against the security mechanism are evaluated. As described in Section 2.1 different groups of attackers are classified. Essentially the first two groups (beginners and independents) should not be able to get access to the device. For the third group, the business, the required investment to get the key or to get access to device internals should be much higher than the expected ROI.

The new security mechanism for a scan chain is evaluated on a system which has a secret key length of 128 bit. From symmetric cipher algorithms is known that a brute force attack with such a high number of different keys is nearly impossible, so that a key length of 128 bit is recommended in [7]. In sum 2^{128} combinations for the secret key are possible which is equal

to 3.4^{38}. Due to statistical analysis the secret key would be found after testing of 1.7^{38} keys in average.

4.8.1 Attacks using a Hardware Trojan

The ASIC which contains the SSCI is relevant to security. The assumption can be done, that the final integration of the device, from writing RTL code to layout and verification of it, is in hand of one company. In many cases an ASIC contains supplied IP cores. These additional cores are delivered in RTL code or in an encrypted netlist. The RTL code can be reviewed and malicious code can be discovered. In case of an encrypted netlist, such a code review is not possible. The developed SSCI in this work is a stand-alone module, which enables and disables the scan chain interface without any interaction to any components. Thus a compromising component, independent whether internal or external, is not security risk. The secret key inside of the OTP is not readable by any other component. Even if the IP core with the Trojan is part of the scan chain, it is not possible to extract useful information to re-enable the scan chain again.

4.8.2 Recovering the Key from the OTP

There are two places where the key is stored, for a certain time interval in the OTP and permanently in the compare logic. In the OTP the key is destroyed after finishing the scan test, so it is vanished. Nevertheless, a device with the secret key stored in the OTP, keeps it secret. A read access is not possible because port for read access is not connected. At the end of the test, the secret key into the OTP is deleted. So no information is in the OTP at the time of shipment to the end user and no useful information can be earned in a reverse engineering step.

In the following is evaluated, whether it is feasible to extract the secret key from the OTP if the step of deleting the secret key during post-production test is skipped.

Optical Inspection In [69] it is described that fuse bits in a layout can be identified. As example in Figure 7 a small regular structure is marked; this is probably the OTP in this ASIC.

For programming the cells a high voltage is required. Either this voltage comes from outside or it is generated by a step up circuit, which is easy to identify in a pure digital layout. From this circuit or the dedicated power pad to the cells a connection exists, so that the OTP can be found by an attacker. The located OTP in the ASIC is a starting point for an attack based on optical inspection.

But recovering the secret key from OTP by optical inspection is infeasible. In Figure 71 several bits of an OTP are shown and every second bit is programmed. Also in this high resolution TEM extended by EDX there is no difference visible between the transistors.

Electrical Probing The approach for getting the secret key from OTP with electrical probing is similar to the compare units. A hole has to be drilled into the ASIC with a FIB at the position of the outputs. The ASIC is connected to the power supply and with a needle probe the value of the OTP can be measured one by one. In contrast to the compare units only one measuring probe is required. For the compare units two different needle probes are necessary. One needle, which injects a low or high signal at the `key_in` and the second needle to grab the output value

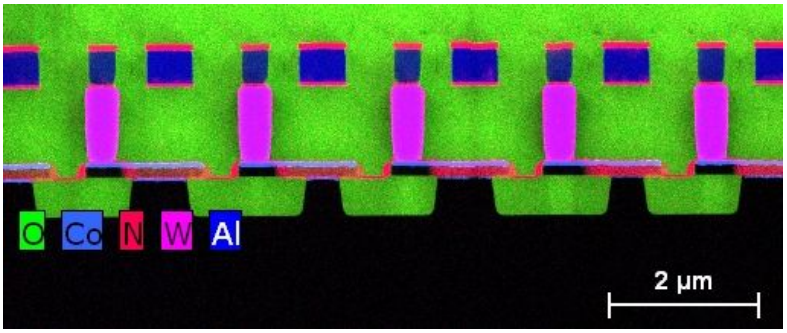


Figure 71: EDX photograph of programmed and unprogrammed OTP cells in IHP $0.25\,\mu m$ technology. A difference between them is not visible.

(a) Top view photograph of vertical metal2 and vias to horizontal metal3. In the red marked area a MIM structure is built-in (200x magnification).

(b) Zoom in of Figure 72a (500x magnification).

(c) Top view of 72a in dark field illumination.

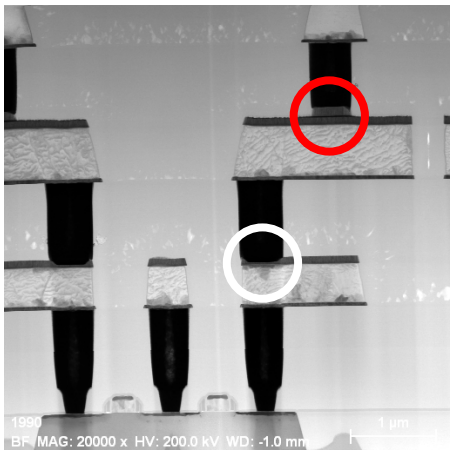
Figure 72: ASIC in IHP $0.25\,\mu m$ technology with MIM structure in optical inspection. Even in the zoom is no difference in top view visible.

of the circuit. In case of the OTP the value is fixed and stored in the memory cell and a stimulus is not required to get the output value.

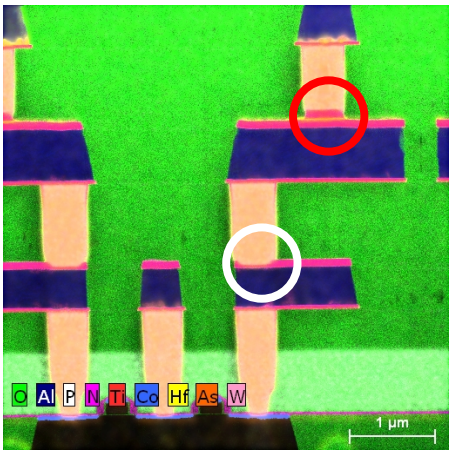
4.8.3 Recovering the Key from the Compare Units

Reverse Engineering For the compare logic the situation is a bit more sensitive. Here the key cannot be erased, i.e. it is still there even after deployment. But the MIM structure used to define the compare cell zero and one cannot be detected by optical inspection. In Figure 72a a photo of a circuit with a MIM structure between the metallization layers 2 and 3 is shown. The marked area in Figure 72b is a via between metal 2 and metal 3 with an additional isolation layer under the via itself. Even in illumination with dark field in Figure 72c no difference between the marked via with MIM structure and all standard vias is visible.

In Figure 72 a top view is shown as it can be seen in a suitable microscope. In contrast to that in Figure 73 is a vertical view of a single structure of a MIM structure is shown. The photograph



(a) TEM photograph of MIM structure in horizontal view between Metal2 and Metal3. The thin isolation is located at the bottom of the view, marked with a red circle. The white circle marks a regular via without isolation.



(b) EDX photograph of 73a. Different materials are marked in different colors. The difference between a via with a connection between two metal layers and the via including the isolation is clearly visible.

Figure 73: MIM structure, fabricated in IHP $0.25\,\mu\mathrm{m}$ technology in horizontal view.

is taken with a Transmission Electron Microscopy (TEM). The additional isolation between the metal 2 and the via on top can be seen, so that the type of the compare unit can be identified.

To generate a photograph as shown in Figure 73 a well-equipped laboratory is necessary. At first with a FIB the part of the compare unit with the MIM structure has to be cut out. Second the piece has to be analyzes in an Energy-Dispersive X-ray spectroscopy (EDX) process. This process has to be done 128 times until all compare units are analyzed. In average one day is required to analyze one compare unit and the DUA is destroyed. After the secret key is extracted, the OTP of a further device has to be fixed to set the key.

Nevertheless such an analysis is required for every compare unit. This means at first all compare units has to be identified in a reverse engineering step and second a very time consuming analysis have to be done.

Electrical Probing A second possibility to extract the secret key from a compare unit is electrical probing. This can be done by micro probing which is normally used for searching defects caused by the production process. The applicability of this technique depends on the manufacturing technology, i.e. gate size, and on the metal layer on which the MIM is used. The lower the layer the more difficult it is to do micro probing. The inputs of the compare unit can be set with a small needle probe. A third needle is placed at the output of the compare unit to read out the value. To perform this attack at first the compare units has to be identified. In the next step a hole has to be grabbed into the ASIC, using a FIB, because the wires for the compare units are on lower level. E.g. for the $0.25\,\mu\mathrm{m}$ IHP technology the width of the wires in metal 2 and 3 is $420\,\mathrm{nm}$, so that a set of very small needle probes is required.

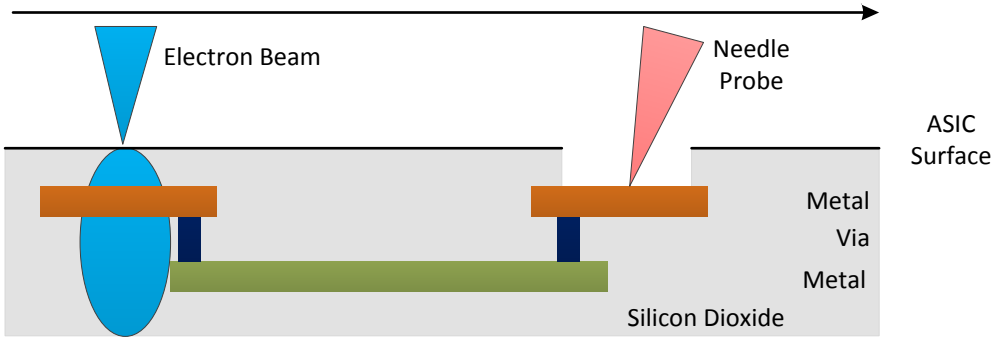


Figure 74: Illustration of the EBAC technique. An electron beam scans the surface of the ASIC and injects electrons and a needle probe is connected to the metal wire.

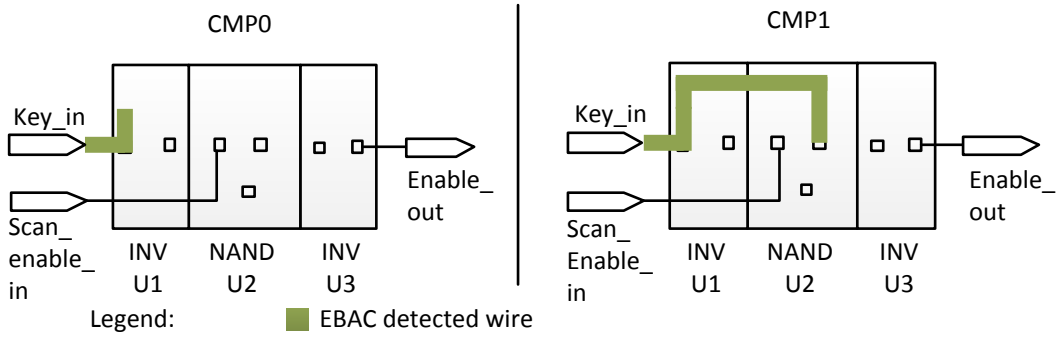


Figure 75: Compare units CMP0 and CMP1 shows different pictures in an EBAC analysis. Only the metal stripe in green is visible.

EBAC An alternative technique to attack the compare logic is the use of a Electron Beam Absorbed Current (EBAC) station. This device is able to detect shorts or opens on metal layers. The principle is shown in Figure 74. The observed metal stripe has to be connected to the EBAC with a needle. An electron beam scans the surface of the ASIC and injects charges. Metal lines absorb the injected charge and a current is induced. This current can be measured with the small needle probe with an amplifier. All connected metal stripes -also on different metal layers- can be seen in the resulting picture.

The wiring of both compare unit types is the same, but the MIM structure causes a different current induction. For this task it is necessary to know the location of all compare units and a FIB to make a hole for connecting the `key_in` signal of every compare unit with the needle probe. Such a system is able to show, whether the compare logic compares to “0” or “1”. The different measurements of the EBAC analysis of both compare units are illustrated in Figure 75.

A relatively simple way to prevent testing of the compare logic by an EBAC system is to implement it on the lower metal layers only, so that its accessibility is hindered by higher metal

Table 13: Calculation of leakage for CMP0. The secret key in OTP is deleted and fixed on logical "1". Only the scan_enable_ can be set.

Cell	*Leakage scan_enable_in="1"*	*Leakage scan_enable_in="0"*
Inverter U1	13.984 pW	14.114 pW
Nand U2	23.350 pW	21.854 pW
Inverter U3	13.984 pW	14.114 pW
Sum	51.319 pW	49.823 pW

Table 14: Calculation of leakage for CMP1. The secret key in OTP is deleted and fixed on logical "1". Only the scan_enable_ can be set.

Cell	*Leakage scan_enable_in="1"*	*Leakage scan_enable_in="0"*
Inverter U1	13.984 pW	13.984 pW
Nand U2	27.968 pW	14.244 pW
Inverter U3	14.244 pW	13.984 pW
Sum	56.198 pW	42.213 pW

layers. Additionally the wiring can be redesigned in a manner, that both compare units have the same view in the EBAC analysis.

Leakage Current Even the leakage current of cells can be helpful for an attacker to get the required information. Leakage current is a property of cells and it cannot be avoided in most cases. A full power-off is the only solution to prevent the very small current flow of the circuit. Although the leakage current has very low entropy it is necessary to verify that even this information is not helpful for an attacker. The goal of the attacker can be to decrease the search space for the secret key by measuring the leakage current. If there is a large difference between the leakage current of both types of compare units, it might be possible to derive the number of CMP0 and CMP1 in a circuit.

In case of a device in which the secret key is deleted, the key_in input of the compare unit is static. The scan_enable_in signal can be set by the user because it is connected to a pad of the ASIC.

In Table 13 and Table 14 the leakage is shown for both types of compares cells. The difference of the leakage between CMP0 and CMP1 is higher, if scan_enable_in signal is set to logical "0". An SSCI with a 128 bit key should have 64 compare units of each type in average. A set of 64 CMP0 and 64 CMP1 units has a leakage of 5.9 nW. A secret key with the lowest leakage consists of 128 CMP1 cells. The leakage is 5.4 nW. In contrast to this, the highest leakage of 6.4 nW has a secret key with 128 CMP0 cells.

So secret key has a leakage current of $5.9\,\mathrm{nW} \pm 0.5\,\mathrm{nW}$. Compared to a processor design of the openMSP430 with a leakage current of 4.3 µW, it is only a fraction. It seems that it is not possible to estimate the number of CMP0 and CMP1 cells in a design. Furthermore the leakage current of a design has a high variation from device to device.

4.8.4 Re-enabling the Scan Chain

The first idea to re-enable the scan chain by an attacker would be to fix the OTP. Keep in mind that it is not sufficient to flip all bits from "1" to "0" using a FIB. This works only for a protection mechanism in which the fuse bit is in the logic path of the `scan_enable` signal. In [58] an example for disconnecting the `scan_enable` signal with one or more fuse bits was discussed. Due to the fact that we use a secret key, every bit of the OTP has to have the right value. Otherwise the scan chain is not enabled.

Since revealing the key directly from the OTP cells or the compare logic is nearly infeasible and due to the fact that no observable operations that could be used as side channel are executed, a potential attacker could use a brute force attack to set the secret key in the OTP.

To repair the memory cells of an OTP special hardware equipment, a FIB, is required. The average time for one bit is approx. 3-4 hours. If the OTP array is 128 bit long and each OTP cell can have the value of "0" or "1", so that 2^{128} different keys are possible. This means that the average time to break an OTP key is $2^{(length\ of\ OTP-1)} * 3.5$ hours. In addition to the time as a prohibitive factor, no ASIC will survive 2^{128} FIB operations.

With an extraction of the secret key from the compare units via FIB or electrical probing, the very high effort of a brute force attack is avoided and the secret key can be set in OTP. Even this approach is time consuming and it is not guaranteed that the device is still working after modification.

An alternative to fix the secret key in the OTP is the modification of the compare units. Every compare unit gets a re-wiring so that the check of the `key_in` signal is bypassed and the `scan_enable` signal is forwarded directly. Even this approach has a high effort because every compare unit has to be fixed with a FIB.

4.8.5 Financial Analysis

In the following a financial analysis is given. The attack is most suitable attack as described in Section 4.8.3. To analyze all compare units at first a reverse engineering step is required to identify them. In a second step every compare unit from the ASIC have to be analyzed in a TEM. The effort, time and costs, for a reverse engineering depends of the design and the facilities in the laboratory and the used software. Once compare unit is identified, it has to be prepared for the analysis in TEM. It has to be cut from the ASIC using a FIB. In service facilities the price for one hour in FIB or TEM is 200 € and the estimated effort in timing is about 20 hours. To analyze one compare unit the costs are 4,000 €.

To protect the scan chain in this work an example with 128 bit is given. It means that for the analysis of the whole secret key a budget of more than 500,000 € has to be calculated. Furthermore 320 working days are required. This estimation includes the analysis of the compare units only without setting the secret key in the OTP. To set one bit in OTP around two working days are required. In case of many metallization layers the time can be increased significantly. Furthermore the device is destroyed completely during the analysis and for fixing the secret key a second device is necessary. This approach works only if the analyzed and the fixed ASIC have the same secret key. In sum an amount of roughly 1,000,000 € seems to be realistic as cost for a potential attack.

For a successful attack a well-equipped laboratory is required and an extensive knowledge in ASIC design and production. Even for business attackers the effort seems very high to re-enabling the scan chain again. An investment of roughly $1{,}000{,}000 €$ is required to get access to the device. The government as attacker class is out of focus because of time, effort and money are no issue. Last but not least it is a very time consuming task to fix the ASIC in that way, that the scan chain is working again.

4.9 Enhancement and Alternative Implementations

The shown mechanism for the SSCI might have some weak points. For example that the secret key is equal for all devices or that the design kit does not offer OTP memory. In the following some enhancements or alternatives for parts of the SSCI are shown.

4.9.1 Configuration and Security of the Secret Key

The secret key to enable the SSCI is part of the design and it is set in an early design phase in the default design process. Many developers have access to the secret key, which can be a security gap. In the age of global supply chains many different companies are involved in the fabrication of an ASIC.

To get a minimum in security, the secret key is stored in an encrypted source file. The key is not readable with a text editor, but from a simulation it is possible to extract it. Unfortunately, the vendors of the design tools for synthesis, simulation and layout do not give any information about the used encryption scheme and it can be that the used algorithm is risky too. Furthermore the encryption scheme is vendor specific. This causes that for synthesis as well as for layout the same vendor of the tools has to be used.

The setting of the secret key should be possible at different levels. Many processes during ASIC fabrication are outsourced and the global supply chain comprises many hands before the ASIC is ready to use. A solution for this problem is a process with trusted suppliers. E.g. the company “Onsemi” offers such a technology. But the additional security increases the cost. Another possibility is to shift the integration of the secret key into the design to a later stage than the RTL design. In Figure 76 the different stages are shown in which the secret key can be set.

In the following the different stages of the device manufacturing are explained, in which a setting of the golden key is possible. As later as in the design flow, as less as persons have access to the golden key. The security of the golden key is the same, equal in which design phase the key is set.

Logic Synthesis During RTL development a source file with a default secret key is used. In the following logic synthesis another file with secret to use key is read-in and the default secret key is replaced by a new one.

Layout The next possible stage to replace a default secret key is in the layout phase. In the configuration step the netlist from the logic synthesis is set as input for the layout tool. It is possible to read in several netlist files, which are combined into one design. This offers the

possibility to read in a netlist with the system design including the SSCI but with a default key. A second netlist contains the component with the "real" secret key and overwrites the default one. The setting of the secret key is moved in a later stage of the design flow and fewer developers have access to the secret key.

Layout Verification During the layout verification process the secret key can be set, too. Therefore it is necessary to use a master design of CMP0 and CMP1 for the layout phase. The difference between both types of compare units is the different placement of small isolators. So, the master for CMP0 and CMP1 consists of the three basic cells without any isolation layer. The isolation layer, the configuration of the compare cells, can be integrated into the back end phase. In the DRC and LVS steps or during generation of layout data for production, a modification of any mask layer is possible. Therefore, a tool can be developed that generates the rectangles in isolation layer to set the configuration of each compare unit. In the end the layout of the ASIC is written in a format, which can be delivered to mask production company.

Fabrication If more security is required and the fab does not trust the mask manufacturer, the secret key can be set even in production. To choose between both types during ASIC production, a change of processing the isolation layer is necessary and a mask is requires which contains a rectangle for isolation at both positions for CMP0 and CMP1. The mask manufacturer does not know which type of compare unit is chosen in the production.

Now a feature of lithographic machines can be used. Selected areas can be exposed, or even not, to create the isolator rectangles individually. An advantage of this solution is that the secret key can be set individually for each ASIC without additional costs for masks set. In this case the effort for the device handling and key management is increased. A database is required, which stores which key is used in which device. Any mistake in this database causes an untestable device and it cannot be shipped to customers.

The fabrication is the last stage in the design flow, in which the type of the compare units, and thereby the secret key, can be chosen. At this point the costumer has to trust the ASIC manufacturer.

4.9.2 Golden Key

Alternative Layout The cost discussed in Section 4.8 holds true only in case the security elements (compare units and OTP) are implemented as described above i.e. a fixed layout. At the moment it based on two inverters and one NAND gate. An implementation which is logical equivalent to the existing solution, but with different types of cells, is also possible. Furthermore the layout of the compare units can be done in many other ways. With low effort the compare units can be prepared in such a way, that the layout is not fixed anymore. The layout tool like Encounter from Cadence [8] places the digital standard cells of the compare units. Typically the cells are not placed side by side, but with some distance between each other and at different positions for each unit. So it is harder for an attacker to identify the compare units in a layout. Such an alternative layout causes more effort and it results in higher cost. To verify the approach of the wiring with MIM layers, it is sufficient to use the fixed CMP units.

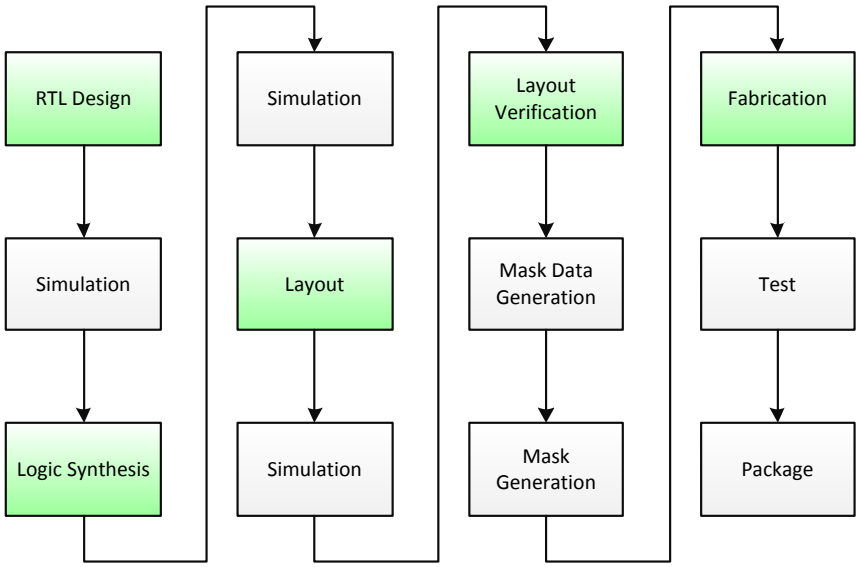


Figure 76: Simplified design flow for ASIC development, production and test. Stages, in which the secret key can be set, are marked in green.

Alternative Storage In principle the compare units are a distributed key storage. It can be replaced by a PUF. After fabrication the value from the PUF is readout, stored in a database and the readout functionality via user interface is disabled in the following. To enable the scan chain the corresponding value for the device from the database is written into its OTP. A component checks the data from the OTP and from the PUF and this replaces the compare units. Such a PUF generates a device specific key and not per design. Furthermore this approach is more secure because no one of the involved parties during the ASIC development and production knows the key and a reverse engineering to get its value is not possible.

Other solutions to store the golden key, like flash or array ROM are not well suitable as described in Section 2.5.2 because they are vulnerable. The value of a flash can be influenced by a laser and the ROM is prone for reverse engineering.

4.9.3 OTP Replacement

For foundries which do not offer OTP memories the OTP can be replaced by SRAM or an flip flop array. In this case before the scan chain test, the secret key has to be written into the memory. The disadvantage of this system is, that it is less secure than the OTP version. The time which is required to brute force a secret key is much smaller than for the OTP version. So it is required to choose a longer key to increase the time for brute forcing significantly. Furthermore the SSCI cannot be fully deactivated after post-production test. While a FIB is necessary to switch the bit value of the OTP, for the volatile memory no special equipment is necessary. Clearly an

OTP bases on a standard CMOS process, so that the circuit can be developed for every CMOS technology.

4.9.4 Interface JTAG

Currently the SSCI is implemented with an SPI as communication interface which is sufficient for a design study. The protection mechanism can be applied on a design with a JTAG interface with the limitation that the SPI is used for the SSCI. To optimize the SSCI it is possible to use the IEEE1500 standard for test interfaces [33]. Besides dedicated registers for controlling the test, such user defined registers are called Core Data Register (CDR). The advantage of this solution is that no further pads are required to set and delete the secret key. Furthermore the IEEE1500 standard allows different subcores, so that different types of cores are possible, e.g. one protected and one unprotected core or two protected cores with different secret keys and so on.

Nevertheless a combination of an unprotected and a protected core should be avoided. It has to be ensured that no secret data can be copied from the protected subcore to the unprotected one. Clearly the unprotected core cannot grab the secret key as described in Section 4.8 but the risk exists that an unprotected core has access to data of the protected core. This is a security leakage.

4.10 Discussion and Comparison

In Table 15 a comparison of different security mechanisms including the SSCI for scan chains is shown. Secure test approaches like BIST are not evaluated because they do not fulfill the requirement of an extensive error diagnosis.

The evaluation of the SSCI showed that the time for activating the scan functionality is in mid-range of all solutions in Table 15. Although it is in the mid-range the SSCI is twenty times faster than the slow ECC based solution. Even the required silicon area is small compared to other solutions. The abandonment of an authentication scheme reduced the implementation effort, area and time for activation significantly.

The public-private key authentication scheme introduced in [18] is effective against attacks, because only the public key is stored into the device. To enable the scan chain a private key has to be known. This private key is not stored in the ASIC and the private key itself is not transferred into the device. In fact the result of a computation is send to the device. This is in contrast to the solution in this work. Here the secret key is stored hidden in compare units. But the effort for the authentication based solution is enormous. The required area is very large, especially in case that an ECC is not needed for the design itself. Furthermore it is very tricky to fulfill the requirements for levels 3 and 4 of FIPS 140-3 [57]. In this document of security requirements for cryptographic modules an exhaustive test of every module is stipulated. But in [18] the ECC module is part of the test infrastructure and a test of this module is not covered. A self-test in operation mode does not fulfill the requirements to get a FIPS 140-3 certificate.

A complex solution with several components often requires a high implementation effort. Every component for the protection mechanism has to be implemented and every additional line

of code requires an extensive test. Otherwise the protected design can get in risk, due to an implementation error.

A public-private key authentication scheme, allowing for the conclusion that the data transferred on the scan interface is encrypted and verified with integrity check. But this protection mechanism is only useful for secured scan interface, which are always on. An interface which is deactivated after the scan chain test and the scan data input and scan data output port are disconnected, does not need such an additional protection. Furthermore an interface, which is enabled during the lifetime of the device, allows a brute-force attack.

The properties of the [19] are similar to the SSCI. The required time for enabling the interface and the area are less and the security is on the same level. The difference between both solutions is the initial phase before testing. For the solution in [19] an enrollment phase is required and PUF value is read of from the device and stored in a database. Further read access to the PUF is prevented by blowing one fuse bit. This means a "raw" device is not protected against access from other users. In contrast to this, the SSCI of a "raw" device cannot be used until the secret key is written into it. This reduces the danger of the misuse of the devices.

Even a very strong authentication based solution or any other protection mechanisms can has a single weak point. Maybe a glitch attack or the modification of a signal wire with a FIB can enable the scan chain interface. A solution with an encrypted scan channel offers more security because a simple bit flip is not enough to establish a data exchange via scan interface. To prevent an attack with a FIB or a needle attack, the solution in this work uses an approach of distributed and connected modules. The required area is much smaller and even the implementation effort is low.

The solution, proposed in this work, is secure against attacker of level 1 and 2 and against attacker of the business type (type 3) until a certain investment. Only with the use of reverse engineering equipment an attack is possible. More secure as this solution is the authentication based system with an encrypted scan channel from [63]. But this solution comes with expensive additional area and very high implementation effort. This increases the cost for the protection mechanism clearly. The higher the cost for a secure scan chain the lower is the probability for its integration.

4.11 Patent Situation

Due to the fact, that the proposed solution in this work is innovative, two patents are pending. Under application number 102014226602.5 and 102014226606.8 both patents are submitted in Germany. One of them describes the compare units and in the second the complete protection scheme is covered.

Different patents exist with respect to OTP memory. E.g. in patent [36] every OTP memory cell has an additional protection cell. Once the protection cell was set, the value of the memory cell cannot be changed anymore. The SSCI uses a secure mechanism for the memory cell too. In contrast to [36] the SSCI has only one protection cell for one data word and it is not fully protected against write access. One type of write access is possible, the deletion of all bits of a data word.

Table 15: Comparison of different solutions for a secure scan chain interface.

Countermeasure	*Implementation Effort*	*Hardware Requirements*	*Area in GE*	*Initial JTAG/SPI Clock Cycles*[1]	*System Clock Cycles*[2]	*Authentication*[3]	*Encrypted Scan Data*	*Integrity Check of Scan Data*	*Permanently Active Interface*	*Enroll Phase*	*Resistant against needle/probing attack*	*Secure against attacker level*
Lock and key technique [11]	+	LFSR	1,551	256	0	yes	no	no	yes	no	no	1
Authentication protocol [61]	–	RNG,SHA-1, EEPROM	124,379	n.s.[4]	n.s.[4]	yes	no	no	yes	no	yes	3
Authentication PUF [19]	–	PUF, fuse	3,094	128	0	yes	no	no	yes	yes	yes	3
Encrypted scan channel (L3) [63]	–	Trivium[83], fuses	126,000	160	1,152	yes	yes	yes	yes	no	yes	3
ChR 0know[18]	–	ECC	46,716	781	482,130	yes	no	no	yes	no	no	1
Single fuse bit	++	Fuse	100[6]	32[7]	1,000[8]	no	no	no	no	no	no	1
SSCI	+	MIM, fuse	5,196	384	24,000[9]	yes	no	no	no	no	yes	3

Legend: ++ very good + good o average - bad – very bad

[1] Number of clock cycles to shift in the additional data to enable the secure scan chain interface.

[2] Number of clock cycles to perform the internal data processing before and after the test, e.g. sequence of authentication protocol or blow a fuse bit.

[3] Authentication between test equipment and device and vice versa.

[4] Not specified. Full authentication scheme, several 1,000 clock cycles expected.

[6] Estimated.

[7] One command to initiate blowing the fuse bit.

[8] Equal to 100 µs at 10 MHz. This time is required for blown fuses in IHP 0.25 µm technology.

[9] 12,000 clock cycles at 10 MHz before test to set the secret key and 12,000 clock cycles to delete the key after post-production test

4.12 Conclusion

In this section a secure solution for the scan chain test was shown. It is highly secure and it allows a full scan chain test without any restrictions. Programs like TetraMax can be used as before to generate scan patterns. The scan test itself has to be enhanced by a setup phase, which writes the secret key in an OTP before the scan test and a delete phase, which deletes the key in the OTP after the scan test. Due to the fact that the OTP for the protection contains many bits and a special compare structure is used, it is almost impossible to attack the device successfully.

The developed schematic and layout for the protection mechanism is one possible implementation. If another set of digital standard cells is available, the circuit would differ. In the end two different logic structures are required which have the functionality of comparing the input with an internal reference value, one for logic "0" and one for logic "1". To avoid an attack by reverse engineering the appearance of both unit types should be the same. Here this problem was solved by using a MIM structure and this is a secure storage of the golden key. The combination of the distributed golden key and the modified `scan_enable` signal prevents a simple needle or probing attack.

Even if some solutions for secure test approaches like an online BIST are simple to implement in RTL, they are not useful for any design. In contrast to this solution, the SSCI requires more changes on different stages in the design flow. Every step is straightforward and the additional work is limited and can be verified. As result a very secure and design-independent protection mechanism with a small area overhead and a small influence on the timing is integrated into the device.

5 Secure Debug Interface

5.1 Requirements

The main requirement of a secure debug interface is that the debug and programming interface cannot be used by attackers. Especially after the programming of the final software release the interface should be fully blocked.

Since scan chains have more or less the same design, the design of the debug interface differs from the once of the scan chain depending of the type of the processor, range of function set or the type of the internal connection between the debug interface and the processor core. It means that the approach for the secure debug interface has to consider these facts. Besides the obvious requirement that the access protection is highly secure, the debug interface should offer full access to the device internals. Even a restricted access, e.g. for a code update without debugging, would be a useful feature of a secure debug interface.

The use of the processor for the authentication protocol to enable the debug functionality is not an option. In case of a trap is required to perform the software-based authentication protocol at the processor. A reset is executed to leave the trap state, but afterwards the register file does not contain the data, which caused the trap, anymore. So, an analysis what was the reason for the trap is not possible.

The software development for sensor nodes is divided in three different stages. The first stage is the development of the software at the desk. The main characteristic is that the device is reprogrammed very often and the debug interface is used extensively. So the additional time for the log-in procedure to enable the programming and debugging should as low as possible. In the next step the devices are in field test. Even in field test a reprogramming and debugging is required to eliminate bugs. In the last step, the final program has to be written into the non-volatile memory. A further reprogramming or debugging should not be possible to ensure that the device is useless for a thief.

A scan chain is used once after fabrication and this is done by the ASIC manufacturer. If all devices have the same secret key, it causes not a problem. But for

In case that all devices have the same secret key to enable the debug interface means that each customer can get access to the devices of all other customers. A group-wise or a device-specific secret key is required to solve this problem. Finally the solution should be resistant against reverse engineering or simple attacks like glitch attacks to prevent non-authorized access to the device.

The requirements for a secure debug interface are:

- Preventing unauthorized use
- Full access to device internals
- Fast access without tedious log-in procedure
- Independent of device reset
- Support of multiple log-in
- Limited access for selected memory areas
- Resistant against reverse engineering
- Group-wise key management

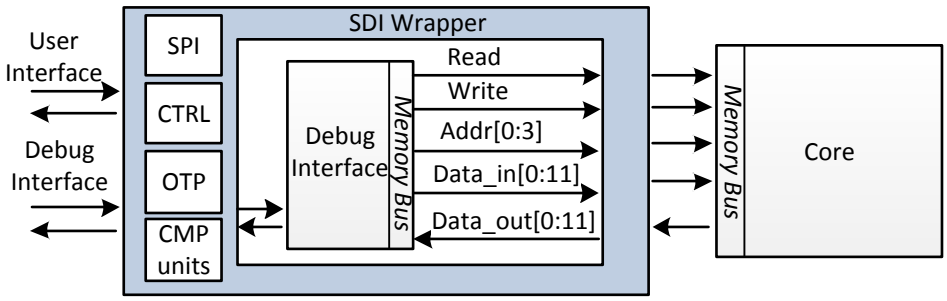


Figure 77: Wrapper for the debug interface, which contains the SDI.

5.2 Secure Debug Interface Concept

In this thesis the proposed approach for the Secure Debug Interface (SDI) bases on an access control as already introduced for the scan chain in Chapter 4. A secret key has to be written into an OTP. For accessing to the debug interface this key has to be identical to the golden key which is stored into the device in compare units. The SDI is adapted to the specific requirements of a debug interface. The SSCI is a wrapper around the whole protected core, for the SDI the wrapper is around the debug component only as shown in Figure 77. The second modification is related to the OTP. The memory gets several entries for the secret key and additional bits for right management.

5.3 Description of the System

In the initial state, it means that the debug interface of the device is never used; the access to the device internals is blocked. To enable the debug interface the user interface of the SDI controller is used to set the access level and the secret key. At first the user chooses the access level. The different access levels are defined by the vendor. For example a full access to the device or programming of the NVM or read access to the processor registers and so on. It is written into the OTP to defined section. Afterwards the secret key is written. Equal to the SSCI every bit of OTP has to have same value as the corresponding CMP unit. In case of one wrong bit value the access to the debug interface is fully blocked. If the secret key is correct the compare units and the following AND module enable the access to the internals of the processor core via the debug unit. Therefore the communication interface of the debug unit and the access to the memory bus of the processor are enabled. To switch off the debug interface a command is sent to the controller of the SDI. This command deletes the secret key and from now on the access to the debug interface is not possible anymore.

The debug unit is not modified during the integration of the SDI and there is no connection to the security mechanism itself. For programming the OTP a second interface is used to isolate the debug functionality and the SDI. This isolation allows an easier integration and increases the security. In the following the SDI is explained in detail.

5.3.1 Wrapper Component

The wrapper combines the user interface, the controller, the OTP and the compare units into a simple module. Furthermore it is designed in such a way that the debug unit is embedded into this unit. It has the advantage that all inputs and outputs of the debug unit can be controlled. In fact the debug unit is a component which has access to the internal memory bus. To implement the protection mechanism all inputs to the debug unit, except `clock` and `reset` signal are blocked and all inputs and outputs to the memory bus are disabled.

5.3.2 Interface

The controller has its own interface to establish the authorization which is fully independent of the debug interface itself. Otherwise a sharing of the resources can cause a security leakage in case of an implementation error. In the example implementation in this thesis an SPI is used. Clearly any other type of interface can be used, e.g. UART, IIC or even 1_wire. An interface with a low pin count is recommended to reduce the additional area.

5.3.3 Compare Units

To secure the debug interface the same type of compare units as described in Section 4 is used. Both types of compare units, CMP0 and CMP1, have the same structure and wiring. The difference is in the placement of the MIM structure. This prevents a fast reverse engineering. To offer a sufficient security a number of 128 compare units are used which is equal to a key length of 128 bit.

5.3.4 Access Level

A very simple approach for an SDI allows an unrestricted access to all registers. In this case no additional bits are required in the OTP for the configuration. If the device vendor supports different access level this information has to be stored in the non-volatile memory. The solution in [61] for a secure scan chain offers an access monitor for different address spaces too. Thus requires an NVM which contains the different addresses to grant the access. A user is able to update this list. From a security point of view this type of protection mechanism seems not to be very secure. An inexperienced user can set the list in a wrong manner. As result the device is not protected any more. An indirect access to all other memory ranges might be possible because a instruction is executable, which copies the data from any address space to another one.

In Figure 78 the solution of this work for SDI with different access levels is shown. The wrapper is located between debug unit and memory bus and the different access levels are defined by different address spaces. In Table 16 an example for the definition of access levels is given. To prevent an attack as described above, the vendor of the processor defines different specific address spaces as access levels. These address spaces are well selected and it is proven that an unrestricted access to other memory ranges is not possible. For example a write-only access to the program memory allows a reprogramming of the device, but inhibits a memory dump or a

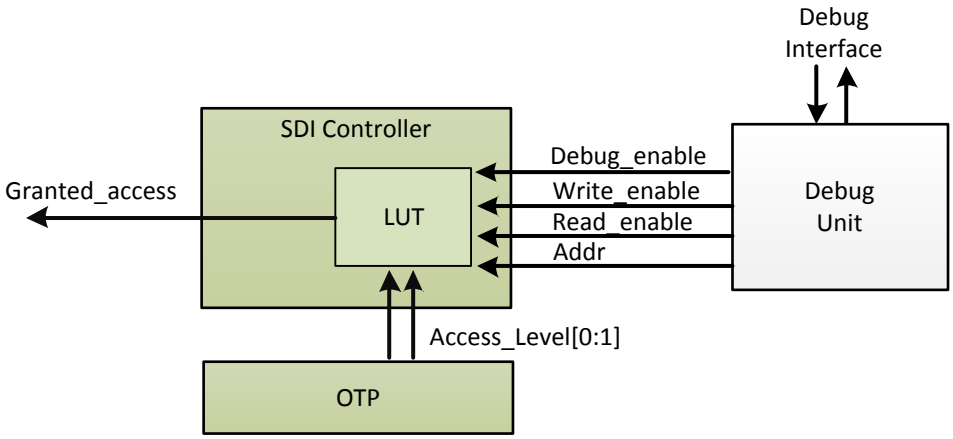


Figure 78: Access level mechanism for SDI. The LUT contains the mapping between access level and allowed address spaces for access. In case of an allowed access the `granted_access` signal enables the connection between debug unit and core. For each request via debug interface the access level is checked.

Table 16: Definition of different access levels on example of the openMSP430.

Access level	*Address range*	*Description*
00	0x0000 – 0xffff	Full access to every address
01	0x0018 – 0x4000	Full access to peripheral only
10	0x8000 – 0xffff	Full access to flash memory only
11	0x8000 – 0xffff	Write access to flash memory only

debugging. Beside this advantage the required area is much smaller. The waiver of an updatable list reduces the silicon area. A small look-up table implemented in combinatorial logic is sufficient and even secure against modification.

5.3.5 OTP

The used type of OTP is similar to the on for the SSCI. The memory block has two interfaces. The typical memory interface is used for writing data. The second one is a full parallel output of the secret key and the corresponding control signals. To support all three stages of software development for a sensor node (desk, field test, final operation), the OTP has three entries for the secret key and the control signals.

In Figure 79 a block diagram of the OTP and the controller is shown. The `key_set` and `key_deleted` signals are used to select the current entry of the secret key. Furthermore the `key_set`, `key_deleted` and `access_level` are status information which is readable via the programming interface of the SDI. It is helpful to know the current state of the SDI but no secret information is forwarded by this read operation.

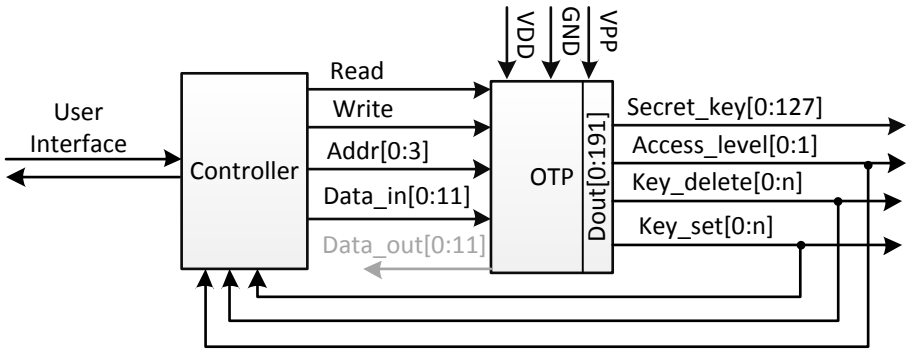


Figure 79: OTP for the SDI. The programming interface on the left side is connected to the controller. The parallel data output on the right side is the input for the compare units and the signals `Access_level`, `Key_deleted` and `Key_set` are forwarded to the controller.

The structure and the usage of the OTP are shown in Tables 17 to 20. The first column "OTP#" specifies the number of the row used in the memory. The `key_set` bit is set after the complete secret key is written into the OTP. Without this bit the secret key is not checked to prevent an attack as described in Section 4.4.3. The `key_deleted` bit allows a fast check whether the secret key is deleted or not. In the `access_level` in this example two bits are stored. One access level defines a full access to the device and the second allows a reprogramming of the flash memory only.

In Table 17 the content of the OTP is given for the first debug session. As long as the key is set the device can be reprogrammed and debugged as many times as needed on the desk. After debugging and programming, the secret key is destroyed by sending the "delete key" command to the SDI controller. The controller deletes the secret key and sets the `key_deleted` bit. Table 18 shows this state after deleting the secret key. In the following Table, Table 19 the content of the memory is shown in case of a further debug session. Finally in Table 20 the OTP is completely filled with logical "1" and a further programming or debugging is not possible.

5.3.6 AND Module

Similar to the SSCI the compare units are connected to an AND module. It ensures that already one wrong bit of the secret key which comes from the OTP causes a blocking of the debug interface. Furthermore the AND module has the task to enable the debug interface for the user in case the correct secret key is provided but to prevent an access to the core if the user does not have access rights for the specific address.

Since the number of inputs of the AND module is fixed, the number of outputs of the unit is flexible. It depends of the number of the signals to protect. At this point the implementation has to be adapted by the user.

Table 17: OTP with programmed first key. The access level is defined with “00”, so that a full access is enabled.

OTP#	*Key_set*	*Key_deleted*	*Access level*	*Key*
1	1	0	00	1010...1011
2	0	0	00	0000...0000
3	0	0	00	0000...0000

Table 18: OTP with deleted first key

OTP#	*Key_set*	*Key_deleted*	*Access level*	*Key*
1	1	1	00	1111...1111
2	0	0	00	0000...0000
3	0	0	00	0000...0000

Table 19: OTP with programmed second key. Only reprogramming of flash memory is allowed, because the access level is defined with “01”.

OTP#	*Key_set*	*Key_deleted*	*Access level*	*Key*
1	1	1	00	1111...1111
2	1	0	01	1010...1011
3	0	0	00	0000...0000

Table 20: OTP all keys are deleted. A further reprogramming or debugging is not possible anymore.

OTP#	*Key_set*	*Key_deleted*	*Access level*	*Key*
1	1	1	00	1111...1111
2	1	1	00	1111...1111
3	1	1	01	1111...1111

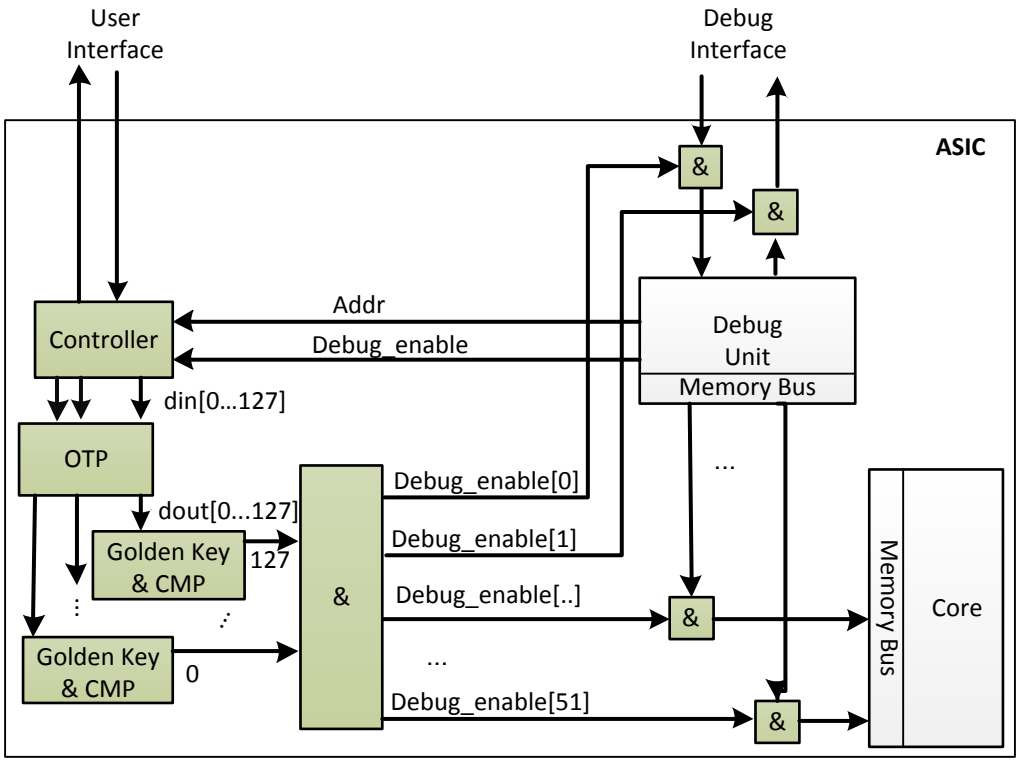


Figure 80: Schematic of the debug interface and protection mechanism. Every signal of the memory bus between the debug unit and the processor core is controlled by an AND gate. Only in the case that the secret key is correct, the communication on the memory bus is possible.

To illustrate this mechanism in the following an example is given. The communication interface (e.g. UART) of the debug unit requires 2 signals. The memory bus of a 16 bit processor has a 16 bit address output, 16 bit data in and 16 bit data out bus and control signals like read enable and write enable, in sum 50 signals. As result to block the 52 signals, the AND module has a 52 bit output as shown in Figure 80. Although the number of outputs of the AND module is less than 128, the length of the secret key is 128 bit. This key length guarantees a high security.

5.4 Evaluation

This section provides the evaluation of the SDI, regarding area, timing and power, bases on the $0.25\,\mu\mathrm{m}$ IHP technology.

Table 21: Area of SDI after synthesis and estimated area for layout.

Component	*Area (synthesis)*	*Area (layout)*[1]
Controller	$0.008\,\mathrm{mm}^2$	$0.011\,\mathrm{mm}^2$
SPI slave	$0.011\,\mathrm{mm}^2$	$0.014\,\mathrm{mm}^2$
Compare cells	$0.013\,\mathrm{mm}^2$	$0.017\,\mathrm{mm}^2$
OTP	- [2]	$0.130\,\mathrm{mm}^2$
Sum		$0.172\,\mathrm{mm}^2$

[1]Estimated
[2]Hard macro, layout only

5.4.1 Area

The SDI has a final layout area of $0.172\,\mathrm{mm}^2$. This is corresponding to $6{,}771$ gate equivalents. A debug unit of the openMSP430 requires $0.089\,\mathrm{mm}^2$ in $0.25\,\mu\mathrm{m}$ IHP technology. It means that the SDI has the double size of the debug unit. The whole system design of the openMSP430 requires $15.8\,\mathrm{mm}^2$. So the additional area for this security mechanism is less than $1\,\%$ of the protected core.

In Table 21 the controller requires a little bit more silicon area as the controller for the SSCI. It supports the different access levels and the check is implemented in the controller. Four additional if clauses, each for every access level, have to be implemented. The estimated overhead in silicon is small. Furthermore the several entries in the OTP require additional area. The given area for the OTP is an estimated value. The memory is built within a memory generator. In the current version it is not possible to generate an OTP as required for an SDI with several entries. The estimation was done on basis of an OTP with one entry plus some area for the additional fuse bits and multiplexers. Compared to the existing components, like address multiplexer for the memory interface and read and write amplifier, the further fuse bits and multiplexers require a small amount of area only.

5.4.2 Timing

The insertion of the security component for the debug interface influences the timing of the system slightly. The speed of the processor core is not influenced. Figure 77 shows that the communication interface and the memory interface of the debug unit are connected to the SDI. Both interfaces can be disconnected with the inserted AND gates as shown in Figure 80. These AND gates influence the timing. For the communication interface a slow serial protocol is used, which is independent of the system clock. The additional delay of $0.09\,\mathrm{ns}$ in the signal path is negligible. For the memory interface the additional delay has more influence. This bus runs at the system clock speed and a significant delay would influence the clock speed. But the additional delay of $0.09\,\mathrm{ns}$ is negligible, too. It can be expected that this has no influence on the system speed, because the openMSP430 is designed as slow and power efficient processor for sensor nodes.

5.4.3 Power

Power in Debug and Programming Mode To enable the debug interface a certain amount of energy is required. The communication interface, the controller and especially the setting and deleting of the secret key in OTP are responsible for that. This step is often performed at the desk. Here the device is powered with a power supply unit.

Power in Operation Mode In the operation mode, i.e. when devices work without any debugging functionality, the SDI consumes a small amount of energy only. The synthesis tools allows an automatic insertion of clock gating, so that a non-used component consumes the leakage current only. For the SSCI component without the OTP a leakage of $24.4\,\mathrm{nW}$ is given. For the OTP the data sheet does not give any information regarding leakage. Compared to the leakage of the openMSP430, i.e. $4.3\,\mu\mathrm{W}$, the leakage of the SDI is negligible.

For a very power sensitive device the debug unit as well as the SDI including OTP can be placed in a separate power island. If these components are not used, they can be disconnected from the power supply and no leakage current floats.

5.5 Adaptation

Different types of debug interfaces and different requirements make it difficult to offer a generic implementation for a secure debug interface. The approach introduced above is an example for a debug unit, which has access to a 16 bit memory bus and the AND module is able to block 52 input and output signals. For another type of memory bus only the AND module has to be adapted to the number of bits to block. Any other component of the SDI can be used unchanged. This simplifies the adaption and increases the security because the controller and the compare units are untouched.

In general the number of entries in the OTP can be adapted to specific requirements. For example for a cost-sensitive mass product it might be sufficient to have one entry in the OTP. This reduces the required area and the cost. Even the different address ranges should be adapted depending on the target. Every processor has its specific registers, different address spaces for volatile and non-volatile memory and so on.

5.6 Security Analysis

From a security point of view, there are two differences between SDI and SSCI. The first one is, that a device unused entries in the OTP for the secret key, e.g. sensor nodes which are distributed in the field. The free entries in the OTP allow an attacker to test so many secret keys as free entries in the OTP are available. Since the number of possible secret keys is 2^{128} ($\approx 3.4 * 10^{38}$) it is nearly impossible that an attacker find the correct secret key with a brute-force attack, even with a high number of devices.

The second difference is regarding the deletion of the secret key. For the SSCI the deletion of the secret key is part of the test sequence. For the SDI the user is free to delete the secret key after a programming and debug session and distribution in field for a test. In case that the secret key is not deleted, it is possibly to use the debug interface as it is fixed in the access level definition.

At this point the developer has to consider this fact and possible to delete the secret key before the device is tested in an environment where access to the device is no longer restricted.

The security analysis in the following is identical to the security analysis for the SSCI. This includes recovering the secret key from the OTP and the golden key from the compare units. Even the re-enabling of the interface with a FIB is very time consuming and risky.

5.7 Discussion and Comparison

In Table 22 a comparison of the developed secure debug interface to other solutions is given. Design and technology independence describe the possibility to transfer the protection mechanism to a new processor and for a new technology. This is important to avoid high cost in a price sensitive area. Afterward the properties for the usage followed by the properties for the security are evaluated. Finally the attacker level for the proposed security mechanism summarizes the table.

Like most of other protection mechanism for the debug interface, the SDI is design independent. If the number of inputs and outputs is changed, a small modification in the AND module has to be done. E.g. the BSL is not design independent, so that for each design a certain implementation effort exists. The core of the SDI is written in HDL and this allows an easy port to a new technology. Only the compare units have to be redesigned. This step has to be done once for a technology and the required effort is low.

A destroyed pad is a solution, which is does not require any additional area and it is dangerous to use as described in Section 3.6.3. Simple fuse bits have a small footprint, but a single element is easy to repair with a FIB. For the additional area overhead to secure the debug interface, the OTP is the dominating factor. Nevertheless compared to the get security the area overhead of rough $1\,\%$ to protect the openMSP430 is negligible. The most expensive solution regarding the area overhead is the module based debug unit. The device has to be fabricated twice. The first ASIC contains the debug unit; the second does not contain it.

The access to the internals of an ASIC which is protected with the SDI does not require an software-based authentication scheme, which would interrupt the software execution. For example the boot loader mechanism needs a reset to execute software which contains the authorization mechanism. A disadvantage of this mechanism is, that after a reset the software routine has to be executed again. Since the SDI stores the secret key in an NVM, the access to the debug interface is enabled after a reset or even power-off. Independent how often the SDI is used, as long as the secret key is not deleted, the SDI is enabled all the time. It is useful for longer debug sessions without frequent input of the secret key. In case of the feature of a permanently on debug interface is not desired, the OTP can be replaced by a volatile memory. To support the different development steps, the SDI can be re-enabled for a predefined number. In case of a blown fuse bit or destroyed pad, such an update is not possible. The RFID based solution and the BSL allows an update at any time. It is not possible to prevent a general access to the device.

The support of different access levels allows an access to the full device or to selected address spaces. This simplifies the maintenance of the device. E.g. for a software update a write access to the flash memory is sufficient and an direct access to processor registers is not possible. This feature is not support by the other protection mechanisms reported in Table 22.

The controller of SDI is not protected against reverse engineering and it would be possible to extract the netlist from the layout. But the controller does not contain secret information so that a reverse engineering of this component is useless. The secret key is stored in the compare units, and as long as the debug interface is enabled, the secret key is stored in the OTP too. The reverse engineering of these components is very time-consuming and expensive. Even the other protection mechanisms for the debug interface are resistant against reverse engineering. For example a blown fuse bit does not allow an access to the device internals. A reverse engineering shows the type of the protection mechanism, but an attacker does not earn secret information from a reverse engineering.

The most important fact of the SDI is the resistance against attacker level 3. It means that even for business attackers the effort for an successful attack is potentially to high. The assumption is under the condition that a very time consuming, and expensive, attack with a FIB is performed. Known attacks, like glitch attacks, are not successful to this protection mechanism. This high protection level offers the SDI and the module based debug unit only. But the module based debug unit has the disadvantage, that a debugging of the final device is not possible. The final device does not contain the debug unit anymore, which makes an update of the program memory impossible. The high security level of the SDI can be supposed because different types of attacks are not successful. For the duration of a probing attack a small needle is connected to an internal wire of the die. E.g. destroyed fuse bit can be bypassed with such a needle. Even a glitch attack is not an option. The protection mechanism for the SDI is independent of the clock signal, so that a clock glitch does not influence the gates.

5.8 Conclusion

The developed protection mechanism for the debug interface in this thesis bases on purposeful destruction of transistors. A 128 bit secret key, stored in an OTP has to be identical to the golden key which is stored in so called “compare units”. These compare units, and even the OTP are resistant against optical reverse engineering. The solution offers a mixture of temporary enabled interface and permanently blocked interface. The different access levels allow a kind of right management. This is sufficient for differentiation of users which have full access to the device and users with a restricted access, e.g. for updating the program memory only.

To enable the debug interface the access level followed by the secret key is written into the device at a separate communication interface. The 128 bit secret key offers a high security and once the interface is fully disabled, only with a very high effort, regarding time and technique, the access to the device can be re-enabled. To disable the interface a “delete command” is sent to the controller and all bits in the OTP are set to “logic 1” in the current entry. A set of entries in the OTP allows a multiple enabling of the debug interface. If all entries in the OTP are deleted an access to the device internals is not possible any more.

The area overhead for the additional component is negligible and the timing of the protected core is influence slightly. Even the additional power consumption is very small and in conjunction with the very high security, the SDI is a smart solution to protect the debug interface of any kinds of microcontrollers.

Table 22: Countermeasures for debug and programming interface and their features and properties. The evaluation based on estimation of the author of this thesis.

Countermeasure	*Design independent*	*Technology independent*	*Implementation effort*	*Area overhead*	*Additional time to initiate debug*	*Reset required before debugging?*	*After reset further debug possible?*	*Software update possible after fabrication?*	*Software update possible after disable?*	*Different access level?*	*Reverse engineering possible*	*Probing attack possible?*[1]	*Glitch attack possible?*	*Leakage problem?* [2]	*Secure against attacker level*
Boot loader (BSL)	no	yes	–	-	yes	yes	no	yes	yes	no	no	no	yes	yes	1
Authentication (RFID)	yes	yes	–	–	yes	no	no	yes	yes	no	no	no	no	no	1
Pad destroying	yes	yes	++	++	no	no	yes	yes	no	no	no	yes	no	no	1
Fuse bits	yes	no	++	++	no	no	yes	yes	no	no	no	yes	no	no	1
Electronic fuse bit	yes	no	++	++	no	no	yes	yes	yes	no	no	yes	no	no	1
Module debug unit	no	yes	++	–[3]	no	no	yes	no	no	no	no	no	no	no	3
SDI	yes[4]	no	++	+	no	no	yes	yes	yes	yes	no	no	no	yes[5]	3

Legend: ++ very good + good o average - bad – very bad

[1] Influence system with needle probe, e.g. force a logic "1" on a specific wire.

[2] One key is valid for many devices and many developers know the secret key.

[3] ASIC has to be fabricated twice. Once with debug unit, once without.

[4] Independent of processor and debug unit. Only a changed debug interface requires an adaption.

[5] If all entries in the OTP are deleted, this is not a risk.

6 Conclusion and Future Work

Recent events have shown that attacks via scan chains or debug interfaces are a common approach to get internal secrets of an ASIC. So countermeasures are indispensable to restrict the access at these interfaces. If a whole sensor network was successfully attacked because the hardware has weak points, all devices have to be replaced causing significant financial effort. Thus in this thesis a secure protection mechanism for scan chain interfaces as well as debug interfaces has been developed.

The security mechanism bases on an authorization scheme with a matching key pair. The user who wants to use the scan chain or debug interface has to know this key and has to write it into the OTP of the ASIC before the device can be tested or debugged. In case that the golden key and the secret key are matching, the access to device internals is enabled. After the post-production test or programming and debugging session the secret key is deleted and the interface is not usable anymore.

The core components developed in this thesis i.e. the adapted logic gates in combination with OTP can be used to secure both types of interfaces. A golden key is stored in the device in a hidden way. A set of 3 digital standard cells are combined in one unit, called CMP. This unit provides one bit of the golden key, either a logical "0" or a "1", and the compare logic. In sum 128 of these units are built in a device and this is representing the golden. The permanent storage for the golden key is typically a weak point due to reverse engineering. In this work, a secure way of storing the key is presented. The approach is to store the golden key in the wiring level of the ASIC. The placement, or even not the placement, of a MIM structure under a via is used to implement a conductive or non-conductive connection between two metal layers. The MIM structure is very tiny and it cannot be seen in a microscope. So the CMP units can store "0" or "1" and optical reverse engineering is prevented.

The properties of the SSCI as well as the SDI regarding area, timing and power consumption are in a range that they can be used for any type of ASIC. Even in the area of wireless sensor nodes, which is a very cost sensitive area, the approach can be used very well. The additional area overhead is less than $1\,\%$ for a typical microcontroller and the power consumption is restricted to the additional leakage of $0.2\,\%$. An advantage of this solution is that it is ready to use and it is a generic solution, completely independent of the type of the system. The solution is simply to integrate, which reduces implementation errors and the scan chain or the debug interface can be used without any limitations. Furthermore it is fully compatible to all design tools, from synthesis, simulation to layout and layout verification, and finally to production. The selected circuits (MIM and OTP memory) are available in many technologies and a device can be fabricated without any further circuit development.

A difference between the scan chain and the debug interface is the point of time of its use. The scan chain is used once after production to check the device for fabrication errors. In contrast to this the debug interface is used for the software development and the programming of the device. The debug interface can be used until the secret key is deleted. The OTP is enhanced to support several entries for the secret key. Each entry can be used to enable the debug interface again. Furthermore a simple access level control is built into the SDI. It grants e.g. full access to all areas in the device or write access to the NVM only.

The security concept is all-encompassing. Once the secret key is deleted, no data can be transferred into or from the core via the protected interfaces. The deletion process uses a physical effect and all 128 anti-fuses are destroyed. This process is not reversible usually. With a FIB it is possible to repair a single element but for 128 elements it is impossible. Furthermore it is not sufficient to repair all anti-fuses. Moreover it has to be set to the correct secret key with a key length of 128 bit. The system is safe against reverse engineering. The units, which contain the golden key, are constructed with digital standard cells. In a digital design it is very difficult to identify these units because they are distributed in the area with all other digital components. The key is stored within the wiring level with a very thin isolation under selected vias. In a typical reverse engineering process with optical inspection the isolation cannot be seen. Finally the mechanism is resistant against micro probing. A dependency between all 128 CMP units causes that only with the correct secret key the interface is enabled. So it is not possible to enable the interface with a single needle. This novel approach is covered by two patents [3].

[3] One patent is granted, one patent is pending.

6.1 Future Work

The approach of this thesis, a mechanism for secure scan chain and debug interface, is well developed. Nevertheless, there are some open issues on system and circuit level and even in technology.

At system level a solution is missing, which supports a secure key programming of a JTAG chain with several devices. All devices in the JTAG chain receive the secret key without encryption. If one ASIC in the chain is owned by an attacker, he has access to the secret key. A type of an encrypted communication channel, from the test environment to every JTAG device, should solve this problem. An asymmetric encryption scheme with a public private key infrastructure for every ASIC is very secure, but the effort regarding additional silicon area is enormous.

The OTP is used in SSCI and SDI and it is clearly to identify during a reverse engineering step. This might be a weak point. In a reverse engineering step the outputs of the OTP can be followed to the connected compare units. An improvement on circuit level would be a single cell for each bit of the OTP. The element has to have the same layout structure as a digital standard cell. This means it can be placed and routed like a digital standard cell. Such an anti-fuse cell is programmed by applying a high programming voltage which is above the breakdown voltage. This voltage is around 6 volts during the typical power supply is around 2.5 volts. The problem comes up that the programming voltage has to be distributed to every OTP cell. This fact has to be solved satisfactory.

Many technologies offer MIM structures, which are used as isolator in this thesis. An optical reverse engineering is possible down to $0.18\,\mu\mathrm{m}$. To extract the schematic for ASICs in smaller technologies equipment like a TEM is necessary. Using this equipment increases the risk that the MIM structures are identified and the golden key can be extracted. Thus a new mechanism for storage of the golden key is required. The functionality should be same which means a small isolation layer but implemented in a way which prevents reverse engineering even with the use of TEM. Especially this point offers high potential for research.

List of Figures

List of Tables

Glossary

ADC	Analogue Digital Converter
AES	Advanced Encryption Standard
ASIC	Application-Specific Integrated Circuit
BSC	Boundary Scan
BIST	Built-In Self-Test
BSL	Bootstrap Loader
CDR	Core Data Register
CMOS	Complementary metal–oxide–semiconductor
CPU	Central Processing Unit
DPA	Differential Power Attack
DES	Data Encryption Standard
DFT	Design for Testability
DRAM	Dynamic Random-Access Memory
DRC	Design Rule Check
DUA	Device under Attack
DUT	Device under Test
EBAC	Electron Beam Absorbed Current
ECC	Elliptic Curve Cryptography
EPROM	Erasable Programmable Read Only Memory
EEPROM	Electrically Erasable Programmable Read Only Memory
EM	Electro Magnetic
EVCD	Extended Value Change Dump
EDX	Energy-Dispersive X-ray spectroscopy
FeRAM	Ferroelectric RAM
FIB	Focused Ion Beamer
FPGA	Field Programmable Gate Array
FTP	File Transfer Protocol
GPIO	General Purpose Input Output
GE	Gate Equivalents
HCI	Hot-Carrier Injection
HDL	Hardware Description Language
IIC	Inter-Integrated Circuit
IP	Intellectual Property
JTAG	Joint Test Action Group
LFSR	Linear Feedback Shift Register
LVS	Layout vs. Schematic
MAB	Memory Address Bus
MAC	Message Authentication Code
MDB	Memory Data Bus
MIM	Metal-Insulator-Metal
MISR	Multiple-Input Signature Register

MKR	Mirror Key Register
MRAM	Magnetoresistive random-access memory
MMU	Memory Management Unit
nmos	n-type metal-oxide semiconductor
NVM	Non-volatile Memory
OTP	One-Time Programmable (Memory)
PCB	Printed Circuit Board
pmos	P-type Metal-Oxide Semiconductor
PRNG	pseudorandom number generator
PUF	Physical Unclonable Function
RAM	random access memory
RFID	Radio-Frequency Identification
RNG	random-number generator
RTL	Register Transfer Level
ROI	Return on Investment
ROM	Read-only Memory
RRAM	Resistive random-access memory
SCA	Side-Channel Analysis
SCP	Secure Copy
SDI	Secure Debug Interface
SoC	System on Chip
SPA	Simple Power Analysis
SPI	Serial Peripheral Interface
SRAM	Static Random-Access Memory
SSCI	Secure Scan Chain Interface
SSL	Secure Socket Layer
STIL	Standard Test Interface Language
SPICE	Simulation Program with Integrated Circuit Emphasis
TAP	Test Access Port
TEM	Transmission Electron Microscopy
TCL	Tool Command Language
UART	Universal Asynchronous Receiver Transmitter
UMTS	Universal Mobile Telecommunications System
USB	Universal Serial Bus
VCD	Value Change Dump
VHDL	Very High Speed Integrated Circuit Hardware Description Language
VM	Volatile Memory
WGL	Waveform Generation Language
WSN	Wireless Sensor Network

References

[1] Adam Laurie. Adams Blog - Obviously a Major Malfunction. http://adamsblogs.aperturelabs.com/2013/01/fun-with-masked-roms.html, 2013. [Online; accessed 02-Sep-2016].

[2] Aeroflex Incorporated. LEON2. "http://gaisler.com/index.php/products/processors".

[3] ATMEL. 8-bit Microcontroller with 32KBytes In-System Programmable Flash ATmega32. http://www.atmel.com/images/doc2503.pdf, 2015. [Online; accessed 27-May-2015].

[4] J. Baukus, L. Chow, and W. Clark. Permanently on Transistor Implemented Using a Double Polysilicon Layer CMOS Process with Buried Contact, July 19 2005. US Patent 6,919,600.

[5] J. Baukus, W. Clark, L. Chow, and A. Kramer. Integrated Circuit Security System and Method with Implanted Interconnections, Feb. 2 1999. US Patent 5,866,933.

[6] F. Beck. *Integrated Circuit Failure Analysis*. Wiley, 1998. Translated by Stephen S. Wilson.

[7] Bundesamt für Sicherheit in der Informationstechnik. Kryptographische Verfahren: Empfehlungen und Schlüssellängen, 2016. BSI TR-02102-1.

[8] Cadence. Encounter . http://www.cadence.com/products/di/edi_system/pages/default.aspx, 2015. [Online; accessed 2015-06-21].

[9] Cadence. Encounter True-Time ATPG . www.cadence.com/products/ld/true_time_test, 2015. [Online; accessed 2015-06-21].

[10] W. C. Carter and P. R. Schneider. Design of Dynamically Checked Computers. In *IFIP congress (2)*, pages 878–883, 1968.

[11] G.-M. Chiu and J. C.-M. Li. A Secure Test Wrapper Design Against Internal and Boundary Scan Attacks for Embedded Cores. *IEEE Trans. VLSI Syst.*, 20(1):126–134, 2012.

[12] L. Chow, J. Baukus, and W. Clark. Integrated Circuits Protected against Reverse Engineering and Method for Fabricating the same Using an Apparent Metal Contact Line Terminating on Field Oxide, July 25 2002. US Patent App. 09/768,904.

[13] F. Courbon, P. Loubet-Moundi, J. Fournier, and A. Tria. Adjusting Laser Injections for Fully Controlled Faults. In E. Prouff, editor, *Constructive Side-Channel Analysis and Secure Design*, volume 8622 of *Lecture Notes in Computer Science*, pages 229–242. Springer International Publishing, 2014.

[14] M. Courtoy. Antifuse NV memory provides superior protection for IoT devices. http://www.electronicproducts.com/Digital_ICs/Memory/Antifuse_NV_memory_provides_superior_protection_for_IoT_devices.aspx, 2015. [Online; accessed 2016-02-01].

[15] J. Daemen, R. Govaerts, and J. Vandewalle. A New Approach to Block Cipher Design. In R. Anderson, editor, *Fast Software Encryption*, volume 809 of *Lecture Notes in Computer Science*, pages 18–32. Springer Berlin Heidelberg, 1994.

[16] J. DaRolt, A. Das, G. D. Natale, M. Flottes, B. Rouzeyre, and I. Verbauwhede. Test Versus Security: Past and Present. *IEEE Trans. Emerging Topics Comput.*, 2(1):50–62, 2014.

[17] J. DaRolt, G. D. Natale, M.-L. Flottes, and B. Rouzeyre. New Security Threats against Chips Containing Scan Chain Structures. In *HOST*, page 110. IEEE Computer Society, 2011.

[18] A. Das. *Differential Scan-based Side-Channel Attacks and Countermeasures : Secure Design-for-Testability (DfT) for Cryptographic Circuits*. PhD thesis, Katholieke Universiteit Leuven, 2013. ISBN 978-94-6018-736-0.

[19] A. Das, Ü. Koçabas, A. Sadeghi, and I. Verbauwhede. PUF-based secure test wrapper design for cryptographic SoC testing. In W. Rosenstiel and L. Thiele, editors, *2012 Design, Automation & Test in Europe Conference & Exhibition, DATE 2012, Dresden, Germany, March 12-16, 2012*, pages 866–869. IEEE, 2012.

[20] Z. Dyka, C. Walczyk, D. Walczyk, C. Wenger, and P. Langendoerfer. Side Channel Attacks and the Non Volatile Memory of the Future. In *Proceedings of the 2012 International Conference on Compilers, Architectures and Synthesis for Embedded Systems*, CASES '12, pages 13–16, New York, NY, USA, 2012. ACM.

[21] E. Ebrard, B. Allard, P. Candelier, and P. Waltz. Review of Fuse and Antifuse Solutions for Advanced Standard CMOS Technologies. *Microelectronic Journal*, 40(12):1755–1765, Dec. 2009.

[22] B. Ege, A. Das, S. Gosh, and I. Verbauwhede. Differential Scan Attack on AES with X-tolerant and X-masked Test Response Compactor. In *15th Euromicro Conference on Digital System Design (DSD), 2012*, pages 545–552, Sept 2012.

[23] B. Frohman. Floating Gate Transistor and Method for Charging and Discharging same, May 2 1972. US Patent 3,660,819.

[24] O. Girard. openMSP430 Microcontroller - Compatible with the Original MSP430 Architecture. "http://opencores.org/project,openmsp430". [Online; accessed 2014-03-05].

[25] T. Goodspeed. Practical Attacks against the MSP430 BSL. In *Twenty-Fifth Chaos Communications Congress (CCC)*, 2008.

[26] B. Gu, T. Coughlin, B. Maxwell, J. Griffiths, J. Lee, J. Cordingley, S. Johnson, E. Karagiannis, and J. Ehrmann. Challenges and Future Directions of Laser Fuse Processing in Memory Repair . In *Semiconductor Equipment and Materials International*, 2003.

[27] L. R. Harriott, A. Wagner, and F. Fritz. Integrated Circuit Repair using Focused Ion Beam Milling. *Journal of Vacuum Science and Technology B*, 4(1):181–184, 1986.

[28] D. Hely, F. Bancel, M.-L. Flottes, and B. Rouzeyre. A Secure Scan Design Methodology. *Design, Automation & Test in Europe Conference & Exhibition*, 1:245, 2006.

[29] D. Hely, F. Bancel, M.-L. Flottes, and B. Rouzeyre. Secure Scan Techniques: A Comparison. In *On-Line Testing Symposium, 2006. IOLTS 2006. 12th IEEE International*, pages 6 pp.–, 2006.

[30] K. N. Henryk Plötz, Jan Krissler. Chiptease. *c't*, 8:80–85, 2008.

[31] S. Henzler. *Power Management of Digital Circuits in Deep Sub-Micron CMOS Technologies*. Springer, 2007. Springer Series in Advanced Microelectronics, Vol. 25.

[32] IEEE 1149.1 Working Group. IEEE Std. 1149.1 - Standard Test Access Port and Boundary-Scan Architecture. http://grouper.ieee.org/groups/1149/1/, 2014. [Online; accessed 30-April-2014].

[33] IEEE 1500 Working Group. IEEE std 1500 - Standard for Embedded Core Test. http://grouper.ieee.org/groups/1500/, 2016. [Online; accessed 2016-01-05].

[34] ISO. Identification Cards – Integrated Circuit Cards – Part 2: Cards with Contacts – Dimensions and Location of the Contacts. http://www.iso.org/iso/home/store/catalogue_ics/catalogue_detail_ics.htm?csnumber=45989, 2013. [Online; accessed 2016-01-06].

[35] J. Kim and K. Lee. 3-Transistor Antifuse OTP ROM Array Using Standard CMOS Process. In *Symposium on VLSI Circuits, 2003. Digest of Technical Papers.*, pages 239–242, June 2003.

[36] W. Kim and Y. Ma. One-time Programmable (OTP) Memory Devices Enabling Programming Based on Protected Status and Methods of Operating Same, Feb. 20 2007. US Patent 7,180,764.

[37] T. Kobayashi, T. Matsue, and H. Shiba. Flip-Flop Circuit with FLT Capability. In *Proc. IECEO Conference*, page 962, 1968.

[38] Y. A. Kotov, S. Y. Sokovnin, and V. A. Skotnikov. Using X-ray Radiation to Erase Information from a CMOS programmable read-only Memory. In *Proceedings of the 12th International Conference on High-Power Particle Beams, 1998. BEAMS '98.* , volume 2, pages 1045–1047 vol.2, 1998.

[39] O. Krause. Entwicklung einer drahtlosen Authentifizierungs- und Updatekomponente für die Debugschnittstelle von Mikrocontrollern. Diplomarbeit, Brandenburgische Technische Universität Cottbus, 2010.

[40] Lawrence E. Bassham III. The Advanced Encryption Standard Algorithm Validation Suite (AESAVS). Technical report, National Institute of Standards and Technology Information Technology, NIST, Laboratory Computer Security Division, 2002. [Online; accessed 30-April-2014].

[41] J. Lee, M. Tehranipoor, C. Patel, and J. Plusquellic. Securing Scan Design Using Lock and Key Technique. In *20th IEEE International Symposium on Defect and Fault-Tolerance in VLSI Systems (DFT 2005), 3-5 October 2005, Monterey, CA, USA*, pages 51–62. IEEE Computer Society, 2005.

[42] J. Lee, M. Tehranipoor, C. Patel, and J. Plusquellic. Securing Designs against Scan-Based Side-Channel Attacks. *IEEE Transactions on Dependable and Secure Computing*, 4(4):325–336, 2007.

[43] S. Library. Circuit Camouflage Technology. http://www.smi.tv/SMI_SypherMedia_Library_Intro.pdf, 2016. [Online; accessed 2016-04-18].

[44] J. Ltd. Product Brief – JN5139. "http://www.farnell.com/datasheets/146662.pdf", 2008. [Online; accessed 2016-06-30].

[45] R. Maes, A. Van Herrewege, and I. Verbauwhede. PUFKY: A Fully Functional PUF-based Cryptographic Key Generator. In *Proceedings of the 14th International Conference on Cryptographic Hardware and Embedded Systems*, CHES'12, pages 302–319, Berlin, Heidelberg, 2012. Springer-Verlag.

[46] Martin Schober. Degate. http://www.degate.org, 2011. [Online; accessed 2015-09-21].

[47] G. Masalskis and R. Navickas. Reverse Engineering of CMOS Integrated Circuits. *Electronics and Electrical Engineering*, 8(88), 2008.

[48] S. Mattke. Passwort gegen Schokolade. "http://heise.de/-3245447", 2016. [Online; accessed 2016-06-30].

[49] E. J. McCluskey and S. Bozorgui-Nesbat. Design for Autonomous Test. *IEEE Trans. Computers*, 30(11):866–875, 1981.

[50] J. McCollum, R. Lambertson, J. Ranweera, J. Moriarta, J.-J. Wang, and F. Hawley. Reliability of Antifuse-Based Field Programmable Gate Arrays for Military and Aerospace Applications. http://klabs.org/richcontent/MAPLDCon01/Papers/B/B4_McCollum_Figures.pdf, 2001. [Online; accessed 2014-05-16].

[51] I. McLoughlin. Reverse engineering of embedded consumer electronic systems. In *IEEE 15th International Symposium on Consumer Electronics (ISCE), 2011* , pages 352–356, June 2011.

[52] Motorola Inc. SPI Block Guide V03.06. http://www.ee.nmt.edu/~teare/ee308l/datasheets/S12SPIV3.pdf, 2000.

[53] D. Mueller. Method of Protecting a Circuit Arrangement for Processing Data. http://www.freepatentsonline.com/6754606.html, 2001. US Patent 6,754,606.

[54] R. Nara, N. Togawa, M. Yanagisawa, and T. Ohtsuki. Scan-based Attack Against Elliptic Curve Cryptosystems. In *ASP-DAC*, pages 407–412. IEEE, 2010.

[55] C. D. Nardi, R. Desplats, P. Perdu, F. Beaudoin, and J. L. Gauffier. EEPROM Failure Analysis Methodology: Can Programmed Charges Be Measured Directly by Electrical Techniques of Scanning Probe Microscopy? In *Proceedings of International Symposium for Testing and Failure Analysis*, ISTFA, 2005.

[56] G. D. Natale, M.-L. Flottes, and B. Rouzeyre. On-Line Self-Test of AES Hardware Implementations. In *DSN - The 37th Annual IEEE/IFIP International Conference on Dependable Systems and Networks*, 2007.

[57] NIST. FIPS PUB 140-3 (Draft): Security Requirements for Cryptographic Modules. http://csrc.nist.gov/publications/PubsDrafts.html#FIPS-140--3, 2009. [Online; accessed 2014-04-30].

[58] H. Palm, M. Smola, and S. Wallstab. Circuit Configuration with Deactivatable Scan Path. http://www.freepatentsonline.com/y2001/0025355.html, September 2001. Patent.

[59] G. Panic. *A Methodology for Designing Low Power Sensor Node Hardware Systems*. PhD thesis, Brandenburgische Technische Universität Cottbus-Senftenberg, 2014.

[60] L. Pierce and S. Tragoudas. Multi-level Secure JTAG Architecture. In *On-Line Testing Symposium (IOLTS), 2011 IEEE 17th International*, pages 208–209, July 2011.

[61] L. Pierce and S. Tragoudas. Enhanced Secure Architecture for Joint Action Test Group Systems. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 21(7):1342–1345, July 2013.

[62] J. Rajendran, M. Sam, O. Sinanoglu, and R. Karri. Security Analysis of Integrated Circuit Camouflaging. In *Proceedings of the 2013 ACM SIGSAC Conference on Computer and Communications Security*, CCS '13, pages 709–720, New York, NY, USA, 2013. ACM.

[63] K. Rosenfeld and R. Karri. Attacks and Defenses for JTAG. *IEEE Design and Test of Computers*, 27(1):36–47, 2010.

[64] M. Ruben. Open JTAG project. "http://www.openjtag.org", 2015. [Online; accessed 2015-05-21].

[65] A. Schubert and W. Anheier. On Random Pattern Testability of Cryptographic VLSI Cores. *Journal of Electronic Testing*, 16(3):185–192, 2000.

[66] G. Sengar, D. Mukhopadhyay, and D. Chowdhury. Secured Flipped Scan-Chain Model for Crypto-Architecture. *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on*, 26(11):2080–2084, Nov 2007.

[67] Y. Shi, N. Togawa, M. Yanagisawa, and T. Ohtsuki. Design for Secure Test - A Case Study on Pipelined Advanced Encryption Standard. In *Circuits and Systems, 2007. ISCAS 2007. IEEE International Symposium on*, pages 149–152, May 2007.

[68] S. Skorobogatov and C. Woods. Breakthrough Silicon Scanning Discovers Backdoor in Military Chip. In E. Prouff and P. Schaumont, editors, *Cryptographic Hardware and Embedded Systems - CHES 2012 - 14th International Workshop, Leuven, Belgium, September 9-12, 2012. Proceedings*, volume 7428 of *Lecture Notes in Computer Science*, pages 23–40. Springer, 2012.

[69] S. P. Skorobogatov. Semi-invasive Attacks – A New Approach to Hardware Security Analysis. Technical Report UCAM-CL-TR-630, University of Cambridge, Computer Laboratory, Apr. 2005.

[70] S. P. Skorobogatov. Local Heating Attacks on Flash Memory Devices. In M. Tehranipoor and J. Plusquellic, editors, *IEEE International Workshop on Hardware-Oriented Security and Trust, HOST 2009, San Francisco, CA, USA, July 27, 2009. Proceedings*, pages 1–6. IEEE Computer Society, 2009.

[71] J.-H. Song, T.-J. Jung, J.-H. Jung, and S.-J. Park. An Efficient Technique to Protect AES Secret Key from Scan Test Channel Attacks. *JSTS: Journal of Semiconductor Technology and Science*, 12(3):286–292, 3 2012.

[72] STMicroelectronics. Dual-core Cortex A9 HMI embedded MPU - Datasheet. http://www.st.com/resource/en/datasheet/spear1340.pdf, 2016. [Online; accessed 2016-07-31].

[73] R. Stogdill. Dealing with Obsolete Parts. *Design Test of Computers, IEEE*, 16(2):17–25, Apr 1999.

[74] A. P. Stroele. Bit Serial Pattern Generation and Response Compaction Using Arithmetic Functions. In *16th IEEE VLSI Test Symposium (VTS '98), 28 April - 1 May 1998, Princeton, NJ, USA*, pages 78–85. IEEE Computer Society, 1998.

[75] Synopsys. Design Compiler. http://www.synopsys.com/Tools/Implementation/RTLSynthesis/Pages/default.aspx, 2015. [Online; accessed 2015-08-21].

[76] Synopsys. DWJTag - TAP Controller with USERCODE Support. http://www.synopsys.com/dw/ipdir.php?c=DW_tap_uc, 2015. [Online; accessed 2015-05-21].

[77] Synopsys. Formality. http://www.synopsys.com/Tools/Verification/FormalEquivalence/Pages/Formality.aspx, 2015. [Online; accessed 2015-03-31].

[78] Synopsys. TetraMAX ATPG - Automatic Test Pattern Generation . https://www.synopsys.com/Tools/Implementation/RTLSynthesis/Test/Pages/TetraMAXATPG.aspx, 2015. [Online; accessed 2015-06-04].

[79] Texas Instruments. eZ430-Chronos Development Tool User's Guide. http://www.ti.com/lit/ug/slau292f/slau292f.pdf, 2013. [Online; accessed 30-April-2014].

[80] Texas Instruments. MSP430 Programming Via the JTAG Interface User's Guide. http://www.ti.com/lit/ug/slau320s/slau320s.pdf, 2015. [Online; accessed 2015-05-07].

[81] Texas Instruments. MSP430F543xA, MSP430F541xA Mixed-Signal Microcontrollers. http://www.ti.com/lit/ds/slas655e/slas655e.pdf, 2015. [Online; accessed 2016-08-04].

[82] Texas Instruments. MSP430 Programming With the Bootloader (BSL). http://www.ti.com/lit/ug/slau319l/slau319l.pdf, 2016. [Online; accessed 2016-08-02].

[83] Y. Tian, G. Chen, and J. Li. On the Design of Trivium. *IACR Cryptology ePrint Archive*, 2009:431, 2009.

[84] F. Vater, R. Barth, P. Langendörfer, and C. Wittke. Halbleiterlogikbauelement mit verschleierter Vergleichslogik, December 2014. Patent.

[85] F. Vater, P. Langendörfer, and C. Wittke. Sicherheitsbauelement zum Zugriffsschutz, December 2014. Patent.

[86] F. Vater and P. Langendörfer. An Area Efficient Realisation of AES for Wireless Devices (Eine flächeneffiziente AES Hardwarerealisierung für drahtlose Geräte). *it - Information Technology*, 49(3):188–, 2007.

[87] B. Yang, K. Wu, and R. Karri. Scan Based Side Channel Attack on Dedicated Hardware Implementations of Data Encryption Standard. In *Test Conference, 2004. Proceedings. ITC 2004. International*, pages 339–344, Oct 2004.

[88] B. Yang, K. Wu, and R. Karri. Secure Scan: a Design-for-Test Architecture for Crypto Chips. In W. H. J. Jr., G. Martin, and A. B. Kahng, editors, *Proceedings of the 42nd Design Automation Conference, DAC 2005, San Diego, CA, USA, June 13-17, 2005*, pages 135–140. ACM, 2005.

[89] B. Yang, K. Wu, and R. Karri. Secure Scan: A Design-for-Test Architecture for Crypto Chips. *IEEE Trans. on CAD of Integrated Circuits and Systems*, 25(10):2287–2293, 2006.

[90] G. Zhang, C. Hu, P. Yu, S. Chiang, S. Eltoukhy, and E. Hamdy. Reliable Metal-to-Metal Oxide Antifuses. In *Electron Devices Meeting, 1994. IEDM '94. Technical Digest., International*, pages 281–284, Dec 1994.