A secure Scan Design Methodology

David Hély[1&2], Frédéric Bancel[1], Marie-Lise Flottes[2], Bruno Rouzeyre [2]
[1] *Smartcard Division, ST Microelectronics Rousset, France*
[2] *LIRMM/UMR 5506 CNRS, Université Montpellier II, France*
david.hely@st.com

Abstract

It has been proven that scan path is a potent hazard for secure chips. Scan based attacks have been recently demonstrated against DES or AES and several solutions have been presented in the literature in order to securize the scan chain. Nevertheless, the different proposed techniques are all ad hoc techniques, which are not always easy to integrate into a completely automated design flow or in an IP reuse environment. In this paper, we propose a scan chain integrity detection mechanism, which respects both automated design flow and IP reuse environment.

1. Introduction

The main hazards induced by the scan path technique into a crypto chip relies on the possibility for hackers to shift out the scan chain content while the circuit contains secret data or data that may be exploited in order to retrieve secret keys [1]. Potential hackers can activate the scan chain by means of two ways. On one hand, the scan path can be enabled thanks to the test controller as a test engineer proceeds during manufacturing testing. On the other hand, the scan path can be directly probed, thanks to the so-called scan chain "brute attack", which consists in applying two probes on the design: one on the scan_enable signal in order to activate the scan path, and the other one on a scan flip-flop output in order to observe the data flow. Scan chain securization should thus address these two main scan chain attacks possibilities. In [2] the authors propose a new test protocol so that activating the scan chain using the test control does not offer the possibility to hackers to unload confidential attack. They also address the scan chain “brute attack” by checking the scan_enable signal integrity in system mode (i.e. while the chip is running secure applications). The scan_enable signal is thus continuously checked in system mode in order to detect any illegal scan chain activation. Once the scan chain activation has been detected while the chip is in system mode configuration, an alarm is generated, which can be used to fully reset the chip for instance. The problem occurring when implementing such a protection relies on the scan_enable signal structure. This signal is created only after logic synthesis, and its structure can change according to design optimisation. This countermeasure requires manual modification of the design database and is then difficult to fully integrate in automated design flow.

In that paper, we present a new scan chain detection method fully compliant with automated design flow.

2. Scan chain shift detection

2.1. Principle

The main purpose of the following countermeasure is to propose a scan chain integrity detector designable at the front-end level. We propose to insert extra flip-flops within the scan path. The current states of these flip-flops are used to check the scan path integrity. The so-called *spy flip-flop* can be integrated within the RTL description of the IP and is further automatically inserted within the scan path during the design synthesis.

Nevertheless, at the frond end level, the scan I/Os of the flip-flops do not exist at the exception of the scan-out output, which is shared with the flip-flop Q output. The detection mechanism has to detect the activation by checking only the functional outputs. At the difference with the scan_enable integrity protection described in [2], the check is done on the data flow and more precisely on the scan-out output of the flip-flop. The spy flip-flop detects thus any data flowing in the scan chain.

2.2. Implementation

In functional mode the flip-flop output depends only of the value of the D input while, once the scan chain is activated the flip-flop output samples data from the scan_in pin. If the functional input of the flip-flop is set to a fixed value (equal to the reset one) the output value only differs from this one when the input being sampled by the flip-flop is the scan one. Thus, a very simple scan chain activation detector consists in inserting such a flip-flop within the scan path. This flip-flop does not require any modification after synthesis, since all the necessary surrounded logic for that mechanism concerns only the D and Q pins of the flip-flop.

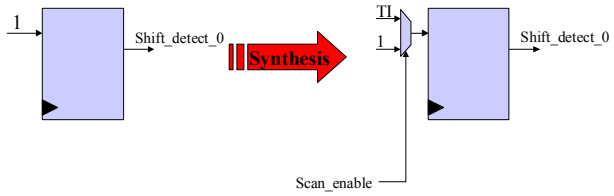

Figure 1: spy flip-flop generation

Fig. 1 illustrates the spy flip-flop generation mechanism. The integration of the shift detection flip-flop into the design is done by adding to the VHDL description of the IP an independent process generating flip-flops having the functional input stuck at 1 or 0. During the design synthesis (incorporating the scan insertion), such a

process generates a D flip-flop having the D input set to a fixed value with its surrounding scan logic. Its input is multiplexed in order to be connected to the scan chain.

In system mode a check is done on the Q output. The check consists in comparing the data being sampled by the flip-flop with the D input. If the comparison indicates a difference, the scan chain activation is detected.. When the flip-flop's input is set to 1 the detection mechanism will then detect a scan chain activation once a 0 flows into the scan chain at this point of the scan chain. In other words, this flip-flop output gives information about the scan_enable signal state. Nevertheless, note that if this signal is 0, then this induces that the scan _enable is at one, the reverse is not always true. The scan chain activation will be detected only when a zero flows on the scan path at this point. Such a detector can easily be integrated into a design without manual intervention during the automatic design flow. The cell is designed at the RT level. During the scan insertion, the flip-flop is replaced by a scan flip-flop and inserted within the scan chain. Spy flip-flops can be designed to detect either 0 or 1 flowing in the scan chain.

Inserting only one spy flip-flop into the scan chain is not sufficient to detect any brute attack on the scan path. Such an attack can only activate a part of the scan chain since the scan_enable signal is bufferized. Moreover, there is a latency time before the scan chain is activated. If by coincidence, the data flow at the spy flip-flop output corresponds to the fixed value of the D input, the detection is not immediate. It is then necessary to spread spy flip-flops within the scan path and to use spy flip-flops detecting both one and zero on the scan path as illustrated in Fig.2.

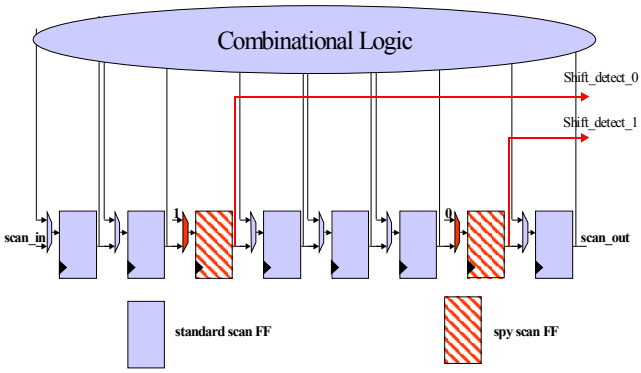

Figure 2: spy cells into the scan path

In order to optimize the design and the insertion of the spy flip-flops within the scan path, spy flip-flops can be easily generated by creating a register bank of spy flip-flops. According to the number of spy flip-flops to be inserted into the scan path, the signal shift_detect is declared as a vector of n bits. Two banks of spy flip-flops are created: shift_detect_0[0, n] and shift_detect_1[0, n]. The top-level rtl view of the scan macro-cell is then modified such that the shift_detect registers are outputted to the security manager or the test controller.

During the scan insertion process, most of synthesis tools allow to specify the scan cell order in the scan path. Thus, the different bits of the two banks of register can be distributed within the scan chain in order to optimize the latency between the probing attack and the detection by the spy flip-flop.

The shift detection controller is a simple combinational network, which can be integrated within the test controller. This module just aims at checking that in case a shift has been detected the test controller is in the correct state. This module inputs the shift_detect signals from the different IPs being scanned. The module outputs for instance a reset to the design so that, once an attack is detected, the circuit is initialized.

3. Conclusion

In order to validate the proposed countermeasure from a security point of view we have performed the scan insertion on a DES IP. 4 shift spy flip-flops have been added to the 198 flip-flop scan path During the validation process, we place a probe on the *scan_enable* signal and observe the data flowing on the scan output pin. Since the detection depends on the data flowing on the scan path, we perform the attack at different time steps of the DES calculation. In the worst case, the detection occurred three clock cycles after the probing attack.

The scan path is slightly increased, which results in an increasing test time and test data volume. Indeed, each shift detection cell added in the scan path increases the scan path length of one new flip-flop. Nevertheless, the bit corresponding to the *shift_detect* cells can be pointed out as don't care ones in order to minimize the impact on memory tester.

Eventually, a global secure scan design methodology can be defined by combining the secure test controller described in [2] with that proposed countermeasure. The two kinds of scan chain attack are then addressed. Moreover, the protection are designed at the front end level with a few synthesis constraints, making the solution adapted to automated design flow and IP reuse methodology.

4. References

[1] B. Yang, K. Wu, R. Karri, "Scan-based Side-Channel Attack on Dedicated Hardware Implementations on Data Encryption Standard", International Test Conference (ITC 2004), Charlottes, USA, October 26-28, pp 339-344

[2] D. Hély, M-L. Flottes, F. Bancel, B. Rouzeyre, "Test Control for Secure Scan Designs", European Test Symposium (ETS 2005), Tallinn, Estonia, May 22-25 2005, pp 190-195