Formal Verification of Secure Reconfigurable Scan Network Infrastructure

Kochte, Michael A.; Baranowski, Rafal; Sauer, Matthias; Becker, Bernd; Wunderlich, Hans-Joachim

Proceedings of the 21st IEEE European Test Symposium (ETS'16) Amsterdam, The Netherlands, 23-27 May 2016

doi: http://dx.doi.org/10.1109/ETS.2016.7519290

Abstract: Reconfigurable scan networks (RSN) as standardized by IEEE Std 1687 allow flexible and efficient access to on-chip infrastructure for test and diagnosis, post-silicon validation, debug, bring-up, or maintenance in the field. However, unauthorized access or manipulation of the attached instruments, monitors, or controllers pose security and safety risks. Different RSN architectures have recently been proposed to implement secure access to the connected instruments, for instance by authentication and authorization. To ensure that the implemented security schemes cannot be bypassed, design verification of the security properties is mandatory. However, combinational and deep sequential dependencies of modern RSNs and their extensions for security require novel approaches to formal verification for unbounded model checking. This work presents for the first time a formal design verification methodology for security properties of RSNs based on unbounded model checking that is able to verify access protection at logical level. Experimental results demonstrate that state-of-the-art security schemes for RSNs can be efficiently handled, even for very large designs.

Preprint

General Copyright Notice

This article may be used for research, teaching and private study purposes. Any substantial or systematic reproduction, re-distribution, re-selling, loan or sub-licensing, systematic supply or distribution in any form to anyone is expressly forbidden.

This is the author's "personal copy" of the final, accepted version of the paper published by IEEE.[1]

[1] **IEEE COPYRIGHT NOTICE**

©2016 IEEE. Personal use of this material is permitted. Permission from IEEE must be obtained for all other uses, in any current or future media, including reprinting/republishing this material for advertising or promotional purposes, creating new collective works, for resale or redistribution to servers or lists, or reuse of any copyrighted component of this work in other works.

Formal Verification of Secure Reconfigurable Scan Network Infrastructure

Michael A. Kochte[1], Rafal Baranowski[1], Matthias Sauer[2], Bernd Becker[2], Hans-Joachim Wunderlich[1]

[1]*ITI, University of Stuttgart, Pfaffenwaldring 47, D-70569 Stuttgart, Germany*
[2]*University of Freiburg, Georges-Köhler-Allee 51, D-79110 Freiburg, Germany*

***Abstract*—Reconfigurable scan networks (RSN) as standardized by IEEE Std 1687 allow flexible and efficient access to on-chip infrastructure for test and diagnosis, post-silicon validation, debug, bring-up, or maintenance in the field. However, unauthorized access or manipulation of the attached instruments, monitors, or controllers pose security and safety risks. Different RSN architectures have recently been proposed to implement secure access to the connected instruments, for instance by authentication and authorization.**

To ensure that the implemented security schemes cannot be bypassed, design verification of the security properties is mandatory. However, combinational and deep sequential dependencies of modern RSNs and their extensions for security require novel approaches to formal verification for unbounded model checking.

This work presents for the first time a formal design verification methodology for security properties of RSNs based on unbounded model checking that is able to verify access protection at logical level. Experimental results demonstrate that state-of-the-art security schemes for RSNs can be efficiently handled, even for very large designs.

***Keywords*-Security, Formal verification, IEEE Std 1687, IJTAG, Reconfigurable scan network, Infrastructure, Side-channel attack**

I. Introduction

Current systems-on-chip and 3D-ICs integrate an increasing amount of infrastructure with different on-chip instruments for test and test control, diagnosis, post-silicon validation, debug, bring-up, or maintenance. Access to this infrastructure is required not only during or after manufacturing, for instance for testing and diagnosis, but also during bring-up and post-silicon validation of the system and in the field, even concurrent to operation for monitoring or reliability management [1, 2, 3].

The growing complexity of on-chip instrumentation requires flexible and scalable access mechanisms. Reconfigurable Scan Networks (RSNs) meet these requirements and have been recently standardized in IEEE Std 1687-2014 [4]. An RSN architecture may use hierarchical gateways called Segment Insertion Bits (SIB), which allow hierarchical control over the accessibility of individual instruments [5]. A SIB is in principle a configurable bypass: it either bypasses the subordinate instrument or sub-network to reduce access time, or connects it to the higher-level scan chain. The mode of operation is chosen by shifting a single configuration bit into the SIB.

The on-chip infrastructure in general and RSNs in particular may constitute a side-channel for attacks, resulting in leakage of confidential information, IP theft, or system manipulation. In complex infrastructures, different access levels and permissions are required for various access scenarios and accessing entities. For instance, during manufacturing test, diagnosis, and post-silicon validation, unlimited access to all instruments should be provided. During power-up system tests in the field, only limited access to certain test features may be sufficient.

The contradicting requirements of providing observability and controllability on the one hand and system security on the other have been recognized and addressed by many researchers [6, 7, 8, 9]. To enforce different access privileges in RSNs, secure architectures have been proposed recently for hierarchical SIB-based RSNs. In these architectures, the SIBs enclosing protected instruments are replaced with *locking* SIBs (LSIB, [10]) or *secure* SIBs (SSIB, [11]). An LSIB is unlocked if a particular secret key is shifted into certain distributed scan elements in the RSNs. The SSIB-based architecture is similar but uses strong cryptographic codes for a challenge-response authentication protocol, so that the secret key is never transmitted in plain text. In [12], an access port protection approach is presented that is applicable to arbitrary RSNs. The recent call for action [13] emphasizes the need for further research on secure design methods for IEEE Std 1687 compliant RSNs.

A large fraction of design bugs are discovered in the infrastructure—a recent industrial study reports that 20% of functional bugs are located in the debug infrastructure itself [14]. Such design bugs may affect protection mechanisms and compromise system security. Ad-hoc and correct-by-construction design methods cannot provide sufficient trustworthiness, and simulation or emulation based validation are inherently incomplete.

To ensure the integrity of secure RSN architectures, unauthorized access must be *proven* impossible. This requires formal techniques to show that certain scan registers are inaccessible to unauthorized users, or that certain access mechanisms are blocked. Formal design verification of modern, and in particular secure, RSNs poses a challenge due to their combinational dependencies and high sequential depth. This renders general formal verification tools for unbounded model checking ineffective [15, 16]. The bounded model checking approach for RSNs in [16] achieves speed-ups of four to six orders of magnitude compared to general tools for unbounded model checking, but can only prove that a certain property holds in a bounded number of time steps. It is not suitable to prove or refute security properties in all reachable states.

In this work, we propose a domain specific unbounded model checking approach to analyze secure RSN architectures. It uses a temporal abstraction in the model of the RSN and Craig interpolants [17] to prove or refute accessibility properties. This method is applied to verify access protection in two different secure RSN architectures.

The following Sections II and III give a brief introduction to the related work and to RSNs in general. Section IV

presents the proposed verification method. The application of the verification on two case studies is described in Section V, followed by the summary of the paper.

II. Related Work

A. Verification of On-chip Access Infrastructure

Verifying the functional correctness of the scan-based access to on-chip instrumentation to avoid design bugs is crucial for rapid bring-up, post-silicon validation, and high product quality. This section reviews the state-of-the-art verification algorithms for connectivity and functionality of scan chains and regular bypass-based scan networks.

The correct connectivity of simple scan chains is checked by structural traversal of the network to find multiple drivers, broken chains, or loop-backs [18]. The accessibility of scan registers in such scan networks requires that a primary input assignment exists such that the scan cells work as a shift register [19].

The structure of an IEEE 1149.1 Test Access Port (TAP) controller and the connectivity of Data Registers can be verified by logic tracing [20]. The functionality of the IEEE 1149.1 circuitry can be validated by four-valued logic simulation using preconditioning and checking sequences [20]. The behavioral rules for IEEE 1500 wrappers can be validated by dynamic, coverage-driven, constrained-random simulation, e.g. [21].

The equivalence of two representations of non-configurable scan networks at different abstraction levels is verified by symbolic simulation [22]. Certain properties of scan networks, such as resetability, can be reduced to combinational equivalence checking [22].

The above-mentioned verification techniques efficiently handle non-configurable scan chains as well as certain regular scan networks with limited configurability, e.g., compliant with IEEE Std 1149.1-2013 or IEEE Std 1500. However, these techniques cannot be directly applied to irregular reconfigurable scan networks with arbitrary control signals and distributed configuration as allowed by IEEE Std 1687.

Combinational and sequential dependencies in RSNs may result in design bugs and cause limited accessibility to scan registers [15] or lead to safety and security issues, e.g., by exposing protected design components. Validation techniques, e.g., simulation or emulation, typically apply non-exhaustive stimuli and hence—particularly with regard to safety and security properties—are insufficient to prove design correctness. Sequential reachability analysis or general model checking methods are required. However, due to the large number of sequential elements and complex combinational and sequential dependencies, such general methods face scalability and robustness issues in RSNs [16], demanding for domain-specific verification approaches for emerging RSN designs.

B. Craig Interpolants in Unbounded Model Checking

Compared to unbounded model checking, bounded model checking (BMC) does not analyze a system for its complete state space, but limits the exploration of its temporal evolution up to a predefined number of time steps. BMC can be used to check if a certain property can be refuted in a sequential circuit, considering a limited number of time steps. If the property can be refuted, an error trace (sequence of state transitions) is generated that leads from an initial state to a state in which the property fails. The behavior of the circuit and the problem conditions are encoded as propositional formula of the form:

$$BMC_k := I_0 \wedge T_{0,1} \wedge \ldots \wedge T_{k-1,k} \wedge \neg P_k \qquad (1)$$

where I_0 encodes the set of initial states. The term $T_{i,i+1}$ represents the transition function, which models the temporal step of the system from time frame i to $i+1$. The predicate P_k represents the goal, i.e., a property whose satisfiability after k steps is to be checked.

The classical BMC procedure first checks whether P fails in the initial state I_0; if not (formula unsatisfiable), it iteratively extends the formula by $T_{i,i+1}$ and checks whether a state with $\neg P$ is reachable in the next step until either $\neg P$ is reachable (formula satisfiable) or the number of steps exceeds the predefined maximum.

BMC is incomplete and cannot prove whether the goal is *unreachable* unless the system is unfolded up to its diameter, which is the smallest number of state transitions to reach all reachable states. Therefore, some approaches attempt to find a fixed point of reachable states instead of enumerating all states until reaching the diameter. Such methods include for instance k-induction [23], BDD-based approaches [24], and methods based on the theory of Craig interpolation [17].

The Craig interpolation based *CIP* solver [25] is used for unbounded model checking in this work since it has been tuned to work well on digital circuit logic and since it provides a model (counter-example) if P is refuted in an instance. CIP is based on McMillan's approach [26] to extend classical BMC to unbounded model checking by employing Craig interpolants. Craig interpolants [17] are used to over-approximate the set of reachable states after a certain number of transition steps. In the CIP solver, this over-approximation is iteratively computed to check if a fixed point of the state space is reached without reaching the goal. In that case, the goal is proven to be unreachable regardless of the number of temporal steps. Additional details of this algorithm can be found in [25].

C. Formal Verification of Security

Commercial verification frameworks for digital circuits allow to prove invariants in the design (assertions) that need to hold. However, for verifying secure infrastructure, such general methods suffer from high sequential depths and aforementioned scalability issues [16]. Furthermore, a substantial manual effort is required to model the specific infrastructure environment as allowed in reconfigurable scan networks.

Other formal approaches specifically provide solutions for analyzing security on different levels. For instance, [27] targets software, or [28] verifies abstract security protocols.

III. Reconfigurable Scan Networks

Reconfigurable scan networks (RSNs) are usually accessed through a JTAG-compliant Test Access Port (TAP).

The RSN can be viewed as a reconfigurable Test Data Register (TDR in IEEE Std 1149.1/JTAG) with *variable length*. The logic state of the RSN determines which scan registers in the network are currently accessible and can be changed by rewriting the content of accessible registers.

The basic building blocks of RSNs comprise scan segments (registers), multiplexers, and combinational logic for complex control and access conditions. IEEE Std 1687 also allows to specify data paths and their control logic.

A *scan segment* is a shift register of one or more bits length with a *select* control input. If *select* is active (segment *is selected*) during a *capture* operation, the shift register is loaded with data from outside of the RSN, e.g. with output of an on-chip instrument. If the segment is selected during a *shift* operation, data is shifted from the segment's scan-input, through its register bits, to the scan-output of the segment. A scan segment may include a *shadow* latch that is stable during the shift operation (as in JTAG test data registers). The optional elements of a scan segment are dashed in Fig. 1a. When the scan segment is selected during an *update* operation, the shadow latch is loaded from the shift register. A scan segment with a shadow latch can be used for bidirectional communication with an on-chip instrument or to drive internal control signals, such as *select* inputs of other scan segments.

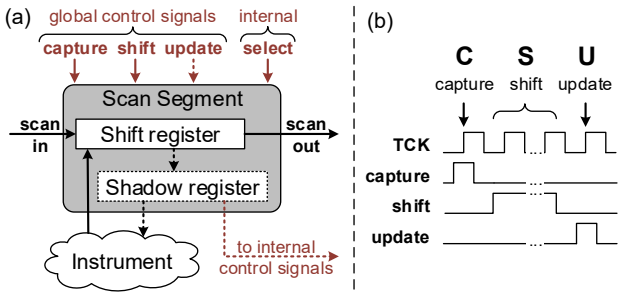

Figure 1. (a) Scan segment; (b) Capture, Shift, Update (CSU) operation

Scan multiplexers control the path through which the data is shifted and can be used, for instance, to bypass scan segments. The *address* control of a scan multiplexer specifies the selected scan input.

The state of the *select* and *address* control signals may depend on the logic state of the RSN itself: These internal control signals may be driven by arbitrary combinational logic that take their input from shadow latches of scan segments or external inputs, which introduces combinational and sequential dependencies between RSN accesses. An RSN has a *primary scan-input* and a *primary scan-output*, as well as global control signals for the capture, shift, and update operations.

An example of an RSN compliant with IEEE Std 1687 is given in Fig. 2. The one-bit scan registers `S1`, `S3` and the external control signal control the access to two multi-bit registers `S2` and `S4`. The scan-in data is shifted through registers `S2` and `S4` only if the previous access assured that $\texttt{S1} = \texttt{S3} = 1$ and the external control signal is asserted.

A *scan path* is a non-circular sequence of connected scan segments starting at a primary scan-in and ending at a primary scan-out port. A scan path is *active* if and only if the select signals for all on-path scan segments are

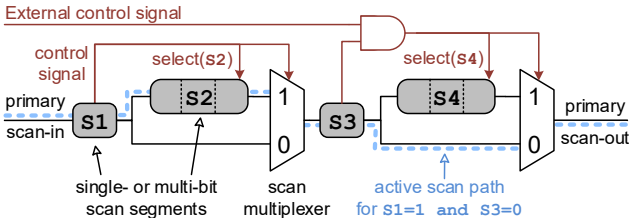

Figure 2. Example of a reconfigurable scan network

asserted, and all on-paths multiplexers address the input that belongs to the active scan path.

A *scan configuration* is the logic state of all sequential elements and external control signals. It is assumed that after reset or power-up all sequential elements are in a known state ('0' or '1'). A scan configuration is *valid* if and only if: (i) one active scan path exists and (ii) scan segments that do not belong to the active scan path are deselected. This ensures that the shift-in data is delivered to the target scan segments, the captured data is shifted towards the primary scan-out, and all scan segments that do not participate in the access (i.e., do not belong to the active scan path) are stable.

The basic RSN access is an atomic operation with three phases: *capture, shift, and update* (CSU, Fig. 1b). During capture, the scan segments on the active scan path may latch new data. Then, this data is shifted out while new scan data is shifted in. Finally, during the update phase, the shifted-in data is latched in the shadow registers on the active scan path. A read or write access to a scan segment requires the accessed segment to be part of the active scan path. A *scan access* is a sequence of CSU operations.

For brevity, we assume that the input *select* of a scan segment enables all the three phases of a CSU operation: If a segment is selected, its state is both captured and updated by the CSU operation. The extension with distinct *capture/update/disable* signals is straightforward [15].

IV. Verification Method

The verification method combines a temporal abstraction for RSNs with unbounded model checking to verify the access protection in secure RSNs. First, the modeling of RSNs is described, followed by the description of the verification method. Section IV-C discusses the capabilities and limitations of the proposed method.

A. CSU-Accurate RSN Model (CAM)

To facilitate efficient formal verification, the sequential complexity of RSNs is reduced by means of temporal abstraction as in [16]. RSNs are represented by a CSU-accurate model (CAM) that is understood as an abstract FSM. A transition in the CAM corresponds to a complete CSU operation and hence covers multiple clock cycles of actual operation. The CAM is constructed from a register-transfer level (RT-level) RSN description in Instrument Connectivity Language (ICL; defined in IEEE Std 1687).

Definition 4.1: The CSU-accurate RSN model $\mathcal{M} = \{S, I, C, c_0, T\}$ consists of a set of state elements S, a set of external control signals I, a set of scan configurations $C \subseteq \{0, 1, X\}^{|S \cup I|}$, the initial scan configuration $c_0 \in C$, and a transition relation $T \subseteq C \times C$. Each state element $s \in S$ corresponds to a

1-bit shadow register of a scan segment in the RSN. A scan configuration $c \in C$ specifies the state of all elements in S and external inputs in I. It is also interpreted as a function $c : S \cup I \to \{0, 1, X\}$ that maps each element $e \in S \cup I$ to state $c(e)$. The transition relation T includes all pairs of scan configurations $(c_1 \in C, c_2 \in C)$ such that c_2 is reachable from c_1 within one CSU operation.

Definition 4.2: The characteristic function of a transition relation T of $\mathcal{M} = \{S, I, C, c_0, T\}$ is defined as:

$$T(c_1, c_2) := \qquad (2)$$
$$\bigwedge_{s \in S} \big[[(\texttt{Active}(c_1, s) = 0) \Rightarrow (c_2(s) = c_1(s))] \wedge$$
$$[(\texttt{Active}(c_1, s) = X) \Rightarrow (c_2(s) = X)]\big],$$

where the predicate $\texttt{Active} : C \times S \to \{0, 1, X\}$ assigns each element $s \in S$ and configuration $c \in C$ a Boolean value which is true ($\texttt{Active}(c, s) = 1$) exactly when s is selected in c and c is a *valid* scan configuration, i.e., when s belongs to the active scan path in c.

Details of the construction of CAMs from RT-level models are given in [16].

The transition relation defines the conditions for state changes in the RSN: if a state element $s \in S$ does not belong to the active scan path in c_1, the state of s does not change between c_1 and c_2. The state of s can only change in a deterministic way if s belongs to the active scan path in c_1 and c_1 is a valid scan configuration. We also assume that the state of s in c_2 becomes unknown (X) when it is unsure whether s belongs to the active scan path in c_1, i.e., when $\texttt{Active}(c_1, s) = X$. Note that for invalid scan configurations, all predicates $\texttt{Active}$ evaluate to X.

The CAM is a sound abstraction: Properties that hold in the CAM are guaranteed to hold in the RT-level RSN model under the assumption that all internal control signals (e.g. multiplexer addresses) are stable during the capture and shift phases [16]. This holds trivially for control signals driven by shadow registers of scan segments, as they may only change during the update operation. The stability of external signals must be guaranteed by the system logic external to the RSN, otherwise the active scan path may change during shifting. The stability of external signals should be formally verified in the RT-level RSN design prior to CAM extraction.

If invalid scan configurations are reachable from the initial scan configuration c_0, the CAM is pessimistic: According to Definition 4.2, the state of a scan segment becomes unknown (X) as a result of a CSU operation if c_1 is invalid. This may be caused if a control or MUX-address signal in the RSN carries an X value, and thus it is unknown which segments are actually active during a CSU operation. Consequently, spurious counter-examples may occur, i.e., some properties that hold for the RSN implementation may not be provable in the CAM. In this case, the CAM based analysis assigns pessimistically X values to the affected segments. If required, a counter-example resulting from an invalid scan configuration can be further investigated by cycle accurate simulation and accurate modeling of X values, for instance using methods proposed in [29]. However, as RSN architectures that allow invalid scan configurations should be avoided anyway to prevent data loss in functional operation [16], the CAM pessimism has little impact on the applicability of the CAM to the verification of security-related properties and allows to find potential design issues early.

B. Verification Method

The verification method is shown in Fig. 3. The inputs are the model of a secure RSN for which the access protection of scan registers or instruments shall be verified at logic level. Commercial tools allow to automatically extract an ICL description from models at gate- or RT-level, or synthesize RT-level models from ICL [30]. The ICL description is then parsed and the CAM is constructed. The transition relation T, which models the possible state transitions by a single CSU operation in the RSN, is extracted as Boolean formula in conjunctive normal form (set of clauses). In addition, the initial (reset) states c_0 of sequential elements in the RSN are identified and encoded as unary clauses. Both sets of clauses are input to the CIP solver.

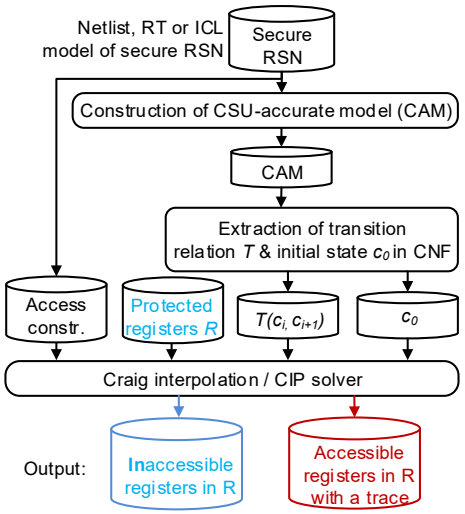

Figure 3. Proposed method for verification of access protection in RSNs.

In a secure RSN, the access to a protected register $r \in R$ shall only be possible if the accessing entity has presented a proof of its authorization, typically by providing a secret key in plain text or encoded in a challenge-response scheme. Here, we are interested if there is a scan access to the RSN that does *not* require to present the correct key, but still performs a read or write access to r. To this end, additional clauses are extracted from the RSN description to model the required constraints, i.e., enforcing a wrong key or secret. The property to be refuted $P := \neg\texttt{Active}(c_k, r)$ expresses that r is not accessible in CSU operation k, i.e., r is protected. To refute this property, the solver searches for a sequence of CSU operations that accesses register r without providing the correct key.

If the CIP solver reaches a fixed point and register r has not been accessible (part of the active scan path) in the intermediate steps, the search concludes. In this case, r is indeed protected and not accessible without providing the key. Otherwise, if r becomes part of the active scan

path, it can be accessed, and a trace or sequence of CSU operations for this case is output.

C. Applicability and Limits

The presented verification method is capable of proving security-related properties under non-invasive attacks that involve controlling of RSN interfaces to external access ports (e.g. JTAG) and to surrounding system logic. For faults that can be fully represented by the CAM (e.g., by modification of the transition relation), resilience to fault-injection attacks can also be verified.

Our approach can be used to verify any property expressed CSU-accurately in terms of the content of shadow registers as well as any data and control signals driven by such registers (either directly or via combinational logic). Since the CAM abstracts from cycle-accurate behavior to state transitions caused by full CSU operations, the intermediate state of scan signals and shift registers cannot be subject to verification. In practice, however, this is no serious limitation as only shadow registers and external control inputs are used for non-invasive access control and hence are relevant for this security verification. The scan signals and shift registers constitute merely an access medium that is assumed to provide arbitrary data to shadow registers whenever the corresponding scan segments belong to the active scan path.

To guarantee the validity of CAM-based verification, the CAM must be a sound abstraction of the RSN. As stated in Section IV-A, it must be formally verified that external RSN control inputs (signals driven by the surrounding system) are stable unless an update operation takes place. This can be handled with state-of-the-art model checkers in an RT- or gate-level RSN model. Additionally, the equivalence of the ICL description (from which the CAM is derived) against the actual RSN implementation must be assured [31]. Although currently no dedicated method for ICL equivalence checking exists, commercial tools for automatic extraction of ICL descriptions from RT- or gate-level models are already available [30].

V. Evaluation

The verification method is evaluated on LSIB- [10] and SSIB-based [11] RSN designs derived from SIB-based benchmarks introduced in [15]. Our method is used to prove the inaccessibility of protected scan registers if the shared secret is not provided by the accessing entity. In addition, effectiveness of the two protection mechanisms is investigated in presence of possibly intentional or unintentional design bugs in the RSN.

In LSIB-based architectures, protected scan segments are enclosed by locking SIBs (LSIBs). An LSIB includes an extra control input which forces bypass operation and hence can render the protected segment inaccessible [10]. To access a protected segment, the corresponding LSIB must be opened by providing a predefined key (secret) to distributed scan segments that control the locking input of the LSIB. We use the proposed method to verify that a protected scan segment is inaccessible as long as the values of the distributed segments does not match the key.

In SSIB-based architectures, scan segments are protected with secure SIBs (SSIBs) that form a *secure* scan chain [11]. The protected segments can only be accessed after a challenge-response authorization procedure is completed and the SSIBs are unlocked via the secure scan chain. As the encryption circuitry used for the authorization protocol is distinct from the RSN, we leave it out in the CAM and model only its interface. We verify that the protected segments are inaccessible if the *responseOK* signal, which determines successful authorization (output of the encryption block), is deasserted. The encryption block can be verified cycle-accurately using conventional model checking techniques.

The structure of the secure RSN is generated as an ICL model. In each RSN, five scan segments are randomly selected and protected by one of the two schemes. In addition to the fault-free designs, we generate mutated ICL descriptions that reflect design bugs (wrong or swapped signal connections, dangling signals), hardware faults, or intentional alterations of the architecture, as described below:

- The reset line of a SIB, SSIB, or LSIB is not connected, or its reset value is inverted.
- The combinational logic on the control/enable signal of an LSIB is mutated by a disjunction with the conjunction of values in other RSN scan registers.
- The *responseOK* signal in the SSIB architecture is mutated by the disjunction with the conjunction of values in other RSN scan registers.

The last two mutations may result in RSNs in which protected registers become accessible without requiring the correct key. However, a non-exhaustive simulation based validation is in principle unable to uncover such cases.

The mutation experiment is repeated ten times per circuit and mutation type. The location of the mutation is selected randomly in the ICL model. Then, the inaccessibility of the five protected scan registers is analyzed.

The framework is implemented in C++ and executed on a single core of an Intel Xeon CPU X5680 (3.33GHz). The used memory of the verification does not exceed 2GB.

Table I summarizes the results. For each benchmark RSN, the number of scan registers is given. The total number of register bits ranges from 1416 (u226) up to 97984 bits (p93791). Column 'v' shows the percentage of access protection violations due to the injected mutation in the secure RSN designs. Column 'd' shows the average depth (number of unrolled frames) required by the CIP solver to prove or refute the access property. This number does not necessarily correspond to the number of CSU operations required to refute the property, since the CIP solver overapproximates the reachable states per step to prove the property. Column 't' gives the average runtime in seconds for the access verification to the five protected scan registers. The runtime is in all cases below 1.15 seconds, there are no aborts. Columns 6 to 8 give the respective values for the SSIB-based designs.

The maximum number of clauses of the transition function is 33547 for the SSIB-based RSN p93791 and the

Table I: CIP-BASED ACCESS PROTECTION VERIFICATION RESULTS

RSN	#Scan	LSIB			SSIB		
	Regs.	v [%]	d	t [s]	v [%]	d	t [s]
u226	40	14.3	3.0	0.055	33.3	3.9	0.088
d281	50	16.2	3.0	0.060	28.6	3.9	0.094
d695	157	17.1	3.1	0.177	47.6	4.0	0.299
h953	46	16.2	3.0	0.060	28.6	3.9	0.085
g1023	65	16.2	3.0	0.065	16.2	4.2	0.110
f2126	36	13.3	3.0	0.048	23.8	3.8	0.073
q12710	21	12.4	3.0	0.013	10.5	4.2	0.052
p22810	254	18.1	3.1	0.297	42.9	4.1	0.467
p34392	103	17.1	3.2	0.125	4.8	4.7	0.150
p93791	588	19.0	3.1	0.715	47.6	4.1	1.149
t512505	128	19.0	3.1	0.153	33.3	4.0	0.233
a586710	32	12.4	3.0	0.043	23.8	4.1	0.073

mutation of the *responseOK* signal. The maximum depth required by the CIP solver is six for the LSIB designs and eight for the SSIB designs.

The results of these experiments show that the presented verification method is applicable to large RSNs with additional complex dependencies due to the design for security. The very low runtime and stable behavior of the unbounded model checking analysis allows to investigate the access protection of the investigated RSN architectures for a high number of different scenarios.

VI. CONCLUSION

Reconfigurable scan networks as standardized by IEEE Std 1687 (IJTAG) offer flexible and scalable access to on-chip infrastructure. To prevent system attacks via this RSN-based access, secure RSN architectures have been proposed. In this paper, we presented a domain specific unbounded formal verification method for RSNs with combinational and deep sequential dependencies. The method allows to verify access protection to sensitive scan registers or instruments by temporal abstraction and use of Craig interpolation. It is applied to two recent secure RSN architectures to investigate accessibility to protected registers in presence of design bugs. The results show that the method is applicable even to large RSNs and exhibits very low runtimes. This constitutes a first step to formal verification of security properties in on-chip infrastructure.

ACKNOWLEDGEMENTS

This work was supported by the German Research Foundation (DFG) under grant WU 245/17-1 (ACCESS) and further work by the Baden-Württemberg Stiftung (IKT-Sicherheit, SHIVA).

REFERENCES

[1] N. Stollon, *On-Chip Instrumentation: Design and Debug for Systems on Chip*. Springer US, 2011.

[2] E. Larsson and K. Sibin, "Fault management in an IEEE P1687 (IJTAG) environment," in *Proc. IEEE Int'l Symp. Design and Diagnostics of Electronic Circuits Systems (DDECS)*, April 2012, embedded tutorial.

[3] J. Rearick and A. Volz, "A Case Study of Using IEEE P1687 (IJTAG) for High-Speed Serial I/O Characterization and Testing," in *Proc. IEEE Int'l Test Conference (ITC)*, 2006, paper 10.2.

[4] "IEEE Std. 1687-2014 – IEEE Standard for Access and Control of Instrumentation Embedded within a Semiconductor Device," Dec. 2014, IEEE Computer Society.

[5] F. Ghani Zadegan, U. Ingelsson *et al.*, "Access Time Analysis for IEEE P1687," *IEEE Trans. on Computers*, vol. 61, no. 10, pp. 1459–1472, October 2012.

[6] D. Hely, M. L. Flottes *et al.*, "Scan Design and Secure Chip [Secure IC Testing]," in *Proc. IEEE Int'l On-Line Testing Symposium (IOLTS)*, 2004, pp. 219–224.

[7] M. Tehranipoor and C. Wang, *Introduction to Hardware Security and Trust*. Springer, 2011.

[8] J. Da Rolt, A. Das *et al.*, "Test versus Security: Past and Present," *IEEE Trans. on Emerging Topics in Computing*, vol. 2, no. 1, pp. 50–62, Mar. 2014.

[9] R. Karri, F. Koushanfar *et al.*, "Guest editorial special section on hardware security and trust," *IEEE Trans. on CAD of Integrated Circuits and Systems*, vol. 34, no. 6, pp. 873–874, June 2015.

[10] J. Dworak, A. Crouch *et al.*, "Don't Forget to Lock your SIB: Hiding Instruments using P1687," in *Proc. IEEE International Test Conference (ITC)*, 2013, paper 6.2.

[11] R. Baranowski, M. A. Kochte, and H.-J. Wunderlich, "Fine-grained access management in reconfigurable scan networks," *IEEE Trans. on CAD of Integrated Circuits and Systems*, vol. 34, no. 6, pp. 937–946, June 2015.

[12] R. Baranowski, M. A. Kochte, and H.-J. Wunderlich, "Access Port Protection for Reconfigurable Scan Networks," *Journal of Electronic Testing (JETTA)*, vol. 30, pp. 711–723, 2014.

[13] J. Dworak and A. Crouch, "A Call to Action: Securing IEEE 1687 and the Need for an IEEE Test Security Standard," in *Proc. IEEE VLSI Test Symp. (VTS)*, April 2015, pp. 1–4.

[14] S. Kumar, "Industrial challenges of bugs and defects," in *NSF/SRC/DFG Joint Workshop on Bugs and Defects in Electronic Systems: The Next Frontier*, Apr. 2013, Schloss Dagstuhl, Germany.

[15] R. Baranowski, M. A. Kochte, and H.-J. Wunderlich, "Modeling, Verification and Pattern Generation for Reconfigurable Scan Networks," in *Proc. IEEE Int'l Test Conf. (ITC)*, 2012, paper 8.2.

[16] R. Baranowski, M. A. Kochte, and H.-J. Wunderlich, "Reconfigurable Scan Networks: Modeling, Verification, and Optimal Pattern Generation," *ACM Trans. Design Automation of Electronic Systems (TODAES)*, vol. 20, no. 2, pp. 30:1–30:27, 2015.

[17] W. Craig, "Three uses of the Herbrand-Gentzen theorem in relating model theory and proof theory," *J. Symbolic Logic*, vol. 22, no. 3, pp. 269–285, 1957.

[18] R. Fisher, "Method and Apparatus to Check the Integrity of Scan Chain Connectivity by Traversing the Test Logic of the Device," Nov. 2002, US Patent App. 10/300,513.

[19] E. Eichelberger and T. Williams, "A Logic Design Structure for LSI Testability," in *Proc. ACM/IEEE Design Automation Conference (DAC)*, New Orleans, Louisiana, USA, 1977, pp. 462–468.

[20] K. Melocco, H. Arora *et al.*, "A Comprehensive Approach to Assessing and Analyzing 1149.1 Test Logic," in *Proc. IEEE Int'l Test Conference (ITC)*, 2003, pp. 358–367.

[21] I. Diamantidis, T. Oikonomou, and S. Diamantidis, "Towards an IEEE P1500 Verification Infrastructure: A Comprehensive Approach," in *Proc. IEEE Int'l Workshop on Infrastructure IP*, 2005.

[22] H. B. Kamepalli, P. Sanjeevarao, and C.-J. Park, "Scan Chain Verification Using Symbolic Simulation," Patent, 2006, US Patent App. 7,055,118.

[23] M. Sheeran, S. Singh, and G. Stålmarck, "Checking Safety Properties Using Induction and a SAT-Solver," in *Int'l Conference on Formal Methods in Computer-Aided Design*, 2000, pp. 108–125.

[24] J. R. Burch, E. M. Clarke *et al.*, "Symbolic Model Checking for Sequential Circuit Verification," *IEEE Trans. on CAD of Integrated Circuits and Systems*, vol. 13, pp. 401–424, 1994.

[25] S. Kupferschmid, M. Lewis *et al.*, "Incremental Preprocessing Methods for Use in BMC," *Formal Methods in System Design*, pp. 1–20, 2011, 10.1007/s10703-011-0122-4.

[26] K. L. McMillan, "Interpolation and SAT-Based Model Checking," in *Int'l Conference Computer Aided Verification*, 2003, pp. 1–13.

[27] S. Chaki, E. Clarke *et al.*, "Modular verification of software components in C," *IEEE Trans. on Software Engineering*, vol. 30, no. 6, pp. 388–402, June 2004.

[28] C. Cremers, "The scyther tool: Verification, falsification, and analysis of security protocols," in *Computer Aided Verification*, ser. Lecture Notes in Computer Science, A. Gupta and S. Malik, Eds. Springer, 2008, vol. 5123, pp. 414–418.

[29] S. Hillebrecht, M. A. Kochte *et al.*, "Accurate QBF-based Test Pattern Generation in Presence of Unknown Values," in *Proc. Conf. on Design, Automation and Test in Europe (DATE)*, 2013, pp. 436–441.

[30] Mentor Graphics Corporation, "Automation of the IEEE 1687 Standard: Tessent IJTAG," 2014, datasheet. [Online]. Available: www.mentor.com

[31] F. G. Zadegan, E. Larsson *et al.*, "Design, Verification, and Application of IEEE 1687," in *Proc. IEEE Asian Test Symp. (ATS)*, 2014, pp. 93–100.