Security Compliance Analysis of Reconfigurable Scan Networks

Lylina, Natalia; Atteya, Ahmed; Raiola, Pascal; Sauer, Matthias; Becker, Bernd; Wunderlich, Hans-Joachim

Proceedings of the IEEE International Test Conference (ITC'19), Washington DC, USA, 11-15 November 2019

doi: https://doi.org/10.1109/ITC44170.2019.9000114

Abstract: Hardware security adds another dimension to the design space, and more and more attention is paid to protect a circuit against various types of attacks like sniffing, spoofing or IP theft. However, all the efforts for security taken by a designer might be sacrificed by afterwards integrating infrastructure for test, diagnosis and reliability management. Especially, access mechanisms like reconfigurable scan networks (RSNs) may open options for side-channel attacks. Using the presented approach an accurate estimation of reachability properties of all considered benchmarks is provided. The method uses a matrix-based reachability analysis of the original design and the augmented design. The reachability analysis covers complex functional dependencies, caused by configuring a single scan path as well as multiple sequentially activated scan paths through the RSN. This approach adds acceptable runtime to the security verification flow of the design, and shows the designer the introduced possible security violations.

Preprint

Security Compliance Analysis of Reconfigurable Scan Networks

Natalia Lylina[1], Ahmed Atteya[1], Pascal Raiola[2],
Matthias Sauer[2], Bernd Becker[2], Hans-Joachim Wunderlich[1]

[1]ITI, University of Stuttgart, Pfaffenwaldring 47, D-70569 Stuttgart, Germany
kaptsona@iti.uni-stuttgart.de, atteyaad@iti.uni-stuttgart.de, wu@informatik.uni-stuttgart.de
[2]University of Freiburg, Georges-Köhler-Allee 51, D-79110 Freiburg, Germany
raiolap@informatik.uni-freiburg.de, sauerm@informatik.uni-freiburg.de, becker@informatik.uni-freiburg.de

***Abstract*—Hardware security adds another dimension to the design space, and more and more attention is paid to protect a circuit against various types of attacks like sniffing, spoofing or IP theft. However, all the efforts for security taken by a designer might be sacrificed by afterwards integrating infrastructure for test, diagnosis and reliability management. Especially, access mechanisms like reconfigurable scan networks (RSNs) may open options for side-channel attacks.**

Using the presented approach an accurate estimation of reachability properties of all considered benchmarks is provided. The method uses a matrix-based reachability analysis of the original design and the augmented design. The reachability analysis covers complex functional dependencies, caused by configuring a single scan path as well as multiple sequentially activated scan paths through the RSN. This approach adds acceptable runtime to the security verification flow of the design, and shows the designer the introduced possible security violations.

***Keywords*– Reconfigurable Scan Networks, Side-Channel Attacks, Security Validation**

I. Introduction

In complex circuits, embedded instruments are essential for test, diagnosis and reliable operation during the entire life cycle [1]. *Reconfigurable scan networks* (RSNs), standardized by IEEE Std. 1687[2] and 1149-2013[3], offer a flexible and scalable access mechanism to those instruments. Figure 1 demonstrates how such an RSN connects instruments and is fed by a test access port (TAP).

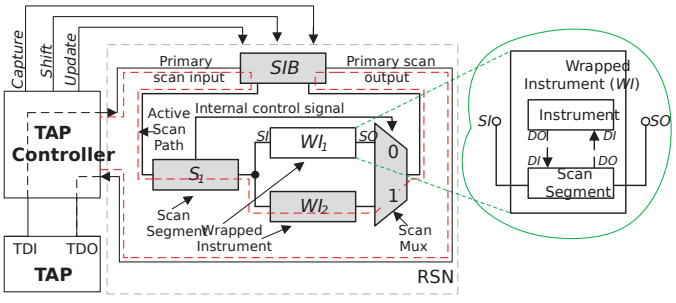

Fig. 1. Example of a Reconfigurable Scan Network (RSN)

Data can be transported between two instruments, which may require a sequence of multiple activated paths. The computation of the control patterns for such a sequence is called retargeting [2]. In many cases, an RSN has to be available during operation for reliability and fault tolerance reasons [1, 4], and restricted rights may be given to an authorized user. An RSN may open various options for side-channel attacks [5, 6] as well as for data and IP manipulation and theft. In-situ sensors may collect internal data of a chip, and other sensors may allow control from outside. In addition, IP cores coming from different sources may not be verified with respect to security. These facts may motivate a system designer to develop the connections in a way that prevents data leakage. Without considering an RSN, the designer may implement such restrictions at system level, for instance, by preventing that certain information is transported from instrument A to instrument B. If a designer ensures that physical connections do not exist between these instruments, the later RSN integration should not introduce such connections again, not even through retargeting. The reachability properties of the initial design must also hold after RSN integration into the design. Information about unwanted data paths, through the RSN, can be obtained either implicitly from circuit description or can be given by system designer explicitly. Significant efforts for protecting RSNs have been spent in recent years [5, 7–10]. IJTAG integrity can be checked using hash-based signatures [11]. An RSN can be augmented with additional registers and scan multiplexers to prevent sniffing and spoofing [12, 13].

In [14], a method is presented to detect and resolve insecure data transfer between two instruments based on a designer-given security specification. The paper at hand goes one step further, and provides a method to verify that the RSN does not add a data path which is not already present in the original circuit. This way, any additional, error-prone security specification for the RSN becomes superfluous. The paper provides a thoughtful analysis of functional and structural dependencies in RSNs and formally identifies the existence of additional data paths between system components. The present approach, is, to the knowledge of the authors, the first to address the impact of complex sequential data dependencies on RSN security by considering the retargeting capabilities.

The goal of the paper is to present a method for the RSN integrator to analyze the access restrictions of the design and to verify the compliance of the RSN structure with these restrictions. If additional dependencies between the design components are introduced through RSN integration, information about possible security violations is generated as an input for later RSN modification. The modifications can be applied either to restrict access patterns using filters [15, 16] or to change the physical connections between scan segments [12, 17] and are not subject of this paper.

The rest of the paper is organized as follows. Section II introduces the basics of RSNs and provides the common terminology. Section III describes how access restrictions and

hence security properties can be extracted from a design. These restrictions can be complemented with additional requirements by a designer. Section IV discusses the approach for reachability analysis, considering complex data and control dependencies of RSNs. Section V shows how the compliance of an RSN with the generated or specified restrictions is verified, and section VI reports experimental results.

II. Reconfigurable Scan Networks

A. Basic Components

The basic components of an RSN are shown in Fig. 1, and comprise the following:

- *Scan Segments* are used to transport the data from a scan input (SI) to a scan output (SO). A scan segment consists of a shift register of a certain length n and an optional shadow register for bidirectional data transfer to the instrument. The operation of a scan segment is driven by external control signals: *Capture*, *Shift*, *Update* and *Select*.
- A *Wrapped Instrument (WI)* includes an instrument, a scan segment and connections to the system through a bidirectional interface. Each instrument, such as aging monitors, Logic or Memory BIST or sensors for internal or external conditions, is connected to the functional part of the system.
- An *Active Scan Path (ASP)* is an acyclic path through *selected* scan segments between primary SI and primary SO. The initial active scan path for this example (Fig. 1) is $SI \rightarrow SIB \rightarrow S_1 \rightarrow WI_2 \rightarrow SO$.
- A *Scan Configuration* is the state of all sequential elements and external control signals. In each valid configuration only one active scan path can exist, and only scan segments on the active scan path are selected.
- *Scan Multiplexers* are used to control the path by choosing one of the scan input branches. The selected scan input can be specified by the *address* control of the multiplexer.
- *Segment Insertion Bits (SIB)* are used to include or bypass some parts of the RSN in the active scan path. If the SIB in Fig. 1 would be closed, the active scan path would be changed to $SI \rightarrow SIB \rightarrow SO$.

B. Capture-Shift-Update-Accurate Model

The basic access to the RSN consists of *capture*, *shift* and *update*-phases. During the *capture*-phase, data is read from the attached instruments. This data is then shifted through the shift registers of selected scan segments during the *shift*-phase. Finally, during the *update*-phase the shifted-in data is latched in the shadow registers. This data can be written to corresponding instruments or used to generate *internal control signals*.

For a set of instruments I and a set of scan segments S of the RSN, the *read*-relation $M^r \subset I \times S$ for each instrument $i \in I$ defines a matching subset of scan segments $S_i^r \subset S$, where any $s^r \in S_i^r$ can directly read data from the corresponding instrument. The *write-relation* $M^w \subset I \times S$ defines for each instrument $i \in I$ a subset of scan segments $S_i^w \subset S$ so that any $s^w \in S_i^w$ can write data into i.

A temporal abstraction as in [18] is used to facilitate the verification of security properties and reduce the sequential complexity of RSNs. *Capture*-, *shift*- and *update*-phases are assumed to form an atomic *CSU-operation*. A sequence of CSU-operations is called *scan access*. In order to read from or write data to a specific scan segment, it must be a part of the *active scan path*. To capture the structural and functional characteristics of the RSN a *CSU-accurate model* is utilized as follows.

Definition 1: The CSU-accurate model (CAM) of an RSN is a tuple $M := \{S, In, C, c_0, T\}$ where the set S represents all state elements, the set In represents external control inputs, the set C denotes possible scan configurations, c_0 represents the initial scan configuration, and T is the transition relation.

Definition 2: The transition relation T of a CAM $M := \{S, In, C, c_0, T\}$ is defined as the set $T \subset C \times C$ that includes all pairs of scan configurations (c_1, c_2) such that $c_2 \in C$ can be reached from $c_1 \in C$ within one CSU-operation.

The *CAM* can be derived from the structural description of an RSN, e.g., from a high-level representation in Instrument Connectivity Language (ICL) [2] or from an RT-level representation. Each transition in the *CAM* corresponds to a complete *CSU*-operation, covering multiple clock cycles. Further details on the *CAM*-model and transition relations are given in [18].

C. An Attack Scenario via RSNs

In Fig. 2 an example is presented, which shows a typical attack scenario, using an RSN as a side-channel. Instruments $i_1, i_2, i_3 \in I$ are being accessed through the RSN in a way that scan segments $s_1, s_2, s_3, temp \in S$ are used to transfer test data. Assume that data transfer between i_1 and i_2 is not possible through the circuit. Consequently, additional dependencies between those instruments should not be introduced by the RSN even through retargeting. In other words, if s_1 is used to capture data from i_1 ($s_1 \in S_{i_1}^r$) and s_2 writes data to i_2 ($s_2 \in S_{i_2}^w$), any data transfer between s_1 and s_2 must be prohibited in the RSN.

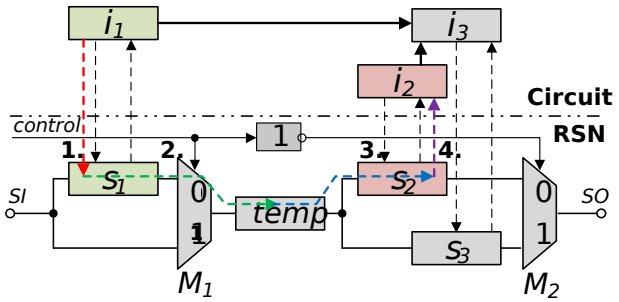

Fig. 2. Additional dependencies due to RSN integration

Even though scan muxes are controlled by external logic signals in a way that no active scan path can include s_1 and s_2 simultaneously, a corresponding dependency is still introduced by retargeting. The attack scenario would consist of the following steps (see numbers in Fig. 2):

1) The data is captured from i_1 to s_1.
2) An attacker configures the RSN in a way that the active scan path through $s_1 \rightarrow temp \rightarrow s_3$ is constructed and shifts data from s_1 to $temp$
3) Having the confidential data in $temp$, an attacker may build a path from $temp$ to s_2 and perform a shift operation.
4) Data from s_2 is updated to i_2.

D. A Graph Model of RSNs

To represent structural and functional dependencies between scan primitives inside the RSN, a graph-based model is constructed (Fig. 3). The RSN is modeled as a directed graph G^{RSN} with the vertex set V^{RSN} and the edge set E^{RSN}, as shown in Fig. 3. Each vertex $v_j^{RSN} \in V^{RSN}$ corresponds to a scan primitive (e.g., *scan segment*, *scan multiplexer*). Models of *SIBs* and other structures are constructed using the same scan primitives.

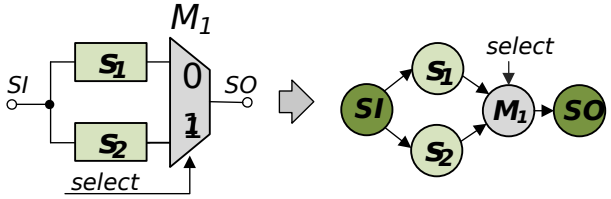

Fig. 3. RSN graph model transformation (select signals for scan segments are omitted for better readability)

The root of the graph is an auxiliary vertex, corresponding to a global scan input, while the sink vertex corresponds to a global scan out. Graph edges $e_j^{RSN} \in E^{RSN}$ represent direct structural connections between scan primitives. Vertices are annotated by logic signals conditions, driving the corresponding *select*-signals of a given primitive.

III. Extraction of Security Constraints

The following section describes the extraction of the security properties from the structural circuit description of the initial design, and optional security requirements specified by the system designer.

The structure of the initial circuit is modeled at Register-Transfer-Level (RTL) by the graph $G^{init} := (V^{init}, E^{init})$ such that each vertex $v^{init} \in V^{init}$ corresponds to a register, whereas edges correspond to connections between registers only through combinational logic blocks CB_1 and CB_2 (Fig. 4).

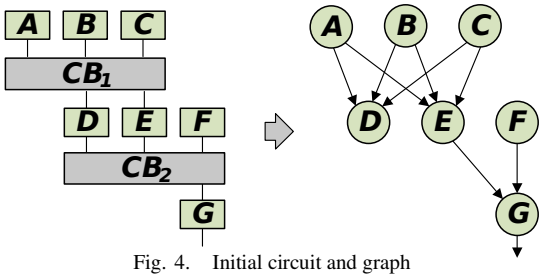

Fig. 4. Initial circuit and graph

G^{init} can be represented by an adjacency matrix $A^{init} \in \mathbb{B}^{|V^{init}|\times|V^{init}|}$ such that an element $a_{k,l}$ equals 1, if a direct edge from vertex v_k to vertex v_l exists, and equals 0, if there is no such edge.

At this point, the matrix A^{init} only describes structural dependencies, but also functional dependencies have to be considered. Assume in Fig. 4, *D* only depends on *A* and *B*, *E* only depends on *B* and *C*, but the combinational block CB_1 is used for logic sharing and area optimization. In this case, there are no dependencies of *C* on *D* and *E* on *A*: $a_{C,D} = a_{A,E} = 0$ In general we can define:

$$r_{k,l} := \begin{cases} 1, & \text{if } v_l \text{ functionally depends on } v_k, \\ 0, & \text{else,} \end{cases} \quad (1)$$

and set $a_{k,l} := a_{k,l} \wedge r_{k,l}$.

The $r_{k,l}$ can be computed by means of false path analysis [19, 20] or by SAT-based methods [14, 21]. We should note that even a false path may propagate glitches or fault attacks [22] and may rise security concerns. The verification of these risks is the duty of the system designer and it is not related to RSN integration.

The transitive closure of the graph G^{init} consists of the nodes V^{init} but contains an edge between $v_a, v_b \in V^{init}$ if there is a path from v_a to v_b in G^{init}. The transitive closure $TCl(A^{init})$ is computed by the algorithm of Warshall and Floyd [23]:

$$R^{init} := TCl(A^{init}) = \bigvee_{l=1}^{|V^{init}|} (A^{init})^l \quad (2)$$

Usually the algorithm converges already for $l \ll |V^{init}|$.

The reachability matrix $R^{init} \in \mathbb{B}^{|V^{init}|\times|V^{init}|}$ covers all the registers of the initial circuit, and not all of them may be connected to an instrument and part of the RSN. Let $V_I \subset V^{init}$ the subset of registers attached to an instrument, and let:

$$R_c^{init} \in \mathbb{B}^{|I|\times|I|} \quad (3)$$

be the submatrix $R_c^{init} \subset R^{init}$, which only has columns and rows related to the instrument set I. We call R_c^{init} the compacted reachability matrix, which describes completely the information transfer between instruments in the original circuit.

IV. RSN Reachability Analysis

In this section, we construct a reachability matrix $R_c^{total} \subset \mathbb{B}^{|I|\times|I|}$ which defines the total data transfer between instruments in the circuits with an integrated RSN. If the difference matrix $R_c^{init} - R_c^{total}$ contains any negative coefficient, an additional information exchange is found and a security violation warning is given. In brief, the following steps are performed:

1) Determine structural pairwise dependencies.
2) Determine the subset of dependencies which belong to a valid scan configuration (assignment of control signals).
3) Determine dependencies between valid scan configurations (retargeting).
4) Extract dependencies corresponding to the instruments.

A. Structural Dependencies

The entire RSN is modeled as a graph $G^{RSN} := (V^P, E^P)$, where the vertices V^P correspond to the scan primitives (scan segments S and scan multiplexers M) of the RSN. The set of all structurally possible connections is an over-approximation of the set of functionally possible connections on the RSN. For the structural reachability, a simple graph traversal is implemented as a preprocessing step for the more time-consuming functional reachability matrix.

From G^{RSN}, the adjacency matrix $A^{RSN} \in \mathbb{B}^{|P|\times|P|}$ is constructed, and similar to (2), the reachability matrix R_{struct}^{RSN} is computed as the transitive closure.

$$R_{struct}^{RSN} := TCl(A^{RSN}) \quad (4)$$

Further, the matrix R_{struct}^{RSN} is reduced to the set of scan segments

$$R_{seg}^{RSN} \in \mathbb{B}^{|S|\times|S|}, R_{seg}^{RSN} \subset R_{struct}^{RSN} \quad (5)$$

If G^{RSN} does not contain cycles, V^P can be topologically sorted and both A^{RSN} and R^{RSN}_{struct} can be rewritten as triangular matrices to reduce runtime. The existence of loops in the RSN can make such a matrix transformation impossible and valid configuration should not activate such a loop. Otherwise, it is considered to be a "bad practice" by IEEE Std. 1687 [2].

B. Reduction to valid scan dependencies

Direct data transfer in an RSN is only possible between *selected* scan segments. The variables $c_1, \ldots c_n$ represent the logic signals, which feed the scan primitives, such as scan segments and scan multiplexers. The logic signals are used to drive the *select* ports of the scan segments and to control, which input of a scan multiplexer is forwarded to its output. For each scan primitive $p_j \in P$ we construct a Boolean formula f_j in *Conjunctive Normal Form (CNF)* to define the *essential select condition* for activating p_j.

Definition 3: The *essential select condition* (ESC) for a given scan primitive $p_j \in P$ is a Boolean formula $f_j(c_1, \ldots c_n)$, defining a group of assignments to logic signal values, required for including this scan primitive to an active scan path.

The scan primitive *ESC* consists of the choice of the succeeding scan segments and the choice of the scan multiplexer input to propagate the data from a given scan primitive to the *scan output*. The second part of the *ESC* is defined by the *relative select condition*.

Definition 4: The *relative select condition* (RSC) $rel_{jk}(c_1, \ldots c_n)$ for a given scan primitive $p_j \in P$ and a scan multiplexer $p_k \in P$, such that p_k is a direct successor of p_j, is a Boolean formula, defining a group of assignments to logic signal values, required to select the specific input of p_k, which is reachable from p_j.

In Fig. 5 an example RSN is shown. The logic control signal c_1 is driven by the shadow register of s_1 and is used to include the scan segment s_2 into the active scan path. The dashed line represents the *ASP* through the scan segments s_1 and s_2.

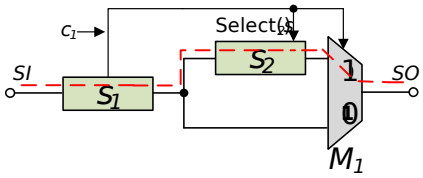

Fig. 5. Logic control signals example

The *ESC* of s_2 should set the select port to 1 ($Select(s_2) = 1$) and activate the 1-input of M_1 (corresponds to the computation of the *RSC*), which means:

$$f_2(c_1) := c_1 \tag{6}$$

An activated scan path has to end in the *scan-out* port, and traversing backward from scan-out we establish the *ESC* for each primitive. In case, a loop is entered this way, the path should lead to unsatisfiable conditions according to [2]. For each $p_j \in P$ *ESC* only depends on the *ESCs* of its direct successors and on the *RSC* of its direct successors and the p_j, and is computed by:

$$f_j(c_1, \ldots c_n) := \bigwedge_{k=1}^{|succ|} (f_k(c_1, \ldots c_n) \wedge rel_{jk}(c_1, \ldots c_n)) \tag{7}$$

For each scan segment, only the control logic signal assignments required to place the scan segment on the active scan path are added to its essential select condition. The logic signals controlling other multiplexers at the active scan path before or after the considered segment are not included in the *ESC*.

C. Active Scan Path Dependencies

Definition 5: The *active scan path (ASP) reachability matrix* R^{RSN}_{path} is a matrix $B^{|S|\times|S|}$, where each element $(r^{RSN}_{path})_{k,l} \in R^{rsn}_{path}$ defines whether data transfer from s_k to s_l is possible through a single active scan path.

The data transfer is possible within one CSU-operation, if a structural path between scan segments exists $((r^{RSN}_{seg})_{k,l} = 1)$ and an assignment to logic signals can be found, such that both scan primitives are selected and an active scan path is formed.

The CNF formulas for scan segments s_k and s_l are combined by conjunction to form a SAT problem instance:

$$path_{k,l}(c_1, \ldots c_n) := f_k(c_1, \ldots c_n) \wedge f_l(c_1, \ldots c_n) \tag{8}$$

with $c_1, \ldots c_n$ being variables to be assigned.

- If the SAT instance is satisfiable, a scan configuration is found, where both scan primitives are selected. The *satisfying assignment* provides the *essential* values of logic signals to put the scan primitives s_k and s_l to the active scan path and makes the corresponding element of the *ASP reachability matrix* equal to 1.
$$(r^{RSN}_{path})_{k,l} := 1, (r^{RSN}_{path})_{k,l} \in R^{RSN}_{path}. \tag{9}$$
- If the SAT instance is unsatisfiable, such a scan configuration does not exist and the conditions for scan primitives are contradicting. The corresponding value in the *ASP reachability matrix* is equal to 0.
$$(r^{RSN}_{path})_{k,l} := 0, (r^{RSN}_{path})_{k,l} \in R^{RSN}_{path}. \tag{10}$$

D. Reachability Matrix Computation

So far we discussed the connectivity via a single activated scan chain. However, multiple reconfigurations may allow an attack scenario as shown in Fig.6. Here, due to the contradicting *select*-conditions of control elements, such as scan multiplexers, data can be propagated from source to destination only between two neighboring scan segments at a time using a single scan configuration. The path through the RSN is depicted in red. The maximum number of reconfigurations, needed to propagate data between two instruments i_1 and i_2, is equal to the sequential depth of the RSN.

Definition 6: The *sequential depth* d of an RSN is the length of the longest topological path inside the RSN.

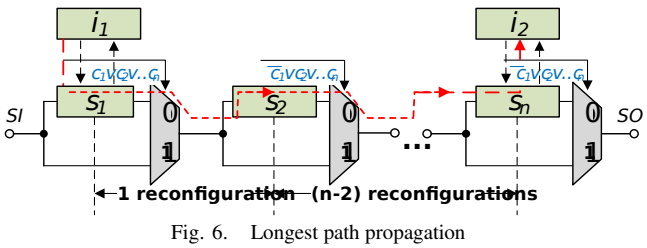

Fig. 6. Longest path propagation

The sequential depth d corresponds to the maximum number of reconfigurations required to transfer data through the RSN [24], which limits the effort to compute the transitive closure of the ASP reachability matrix from (9):

$$R^{RSN} := T\,Cl(R^{RSN}_{path}). \tag{11}$$

Since the CAM-Model is used for RSN reachability analysis, the sequential depth of a given RSN is equal to the highest number of scan segments forming a path. The functional reachability matrix R^{RSN} computation stops, when the computation converges, at most after d iterations.

V. Security Compliance Verification

A. Validation procedure

In Section III we analyzed the possible data flow of the initial circuit R_c^{init}, and in the section above we computed the possible data flow R^{RSN} in the RSN including retargeting. The data flow computation after RSN integration has to also consider the set I of instruments, and an intermediate connectivity matrix $R^{connect} \in \mathbb{B}^{m \times m}$, $m = (|S| + |I|)$ is used, which combines the information about the possible paths between scan segments $s \in S$ with the information, obtained from the *read-* and *write*-relations M_r and M_w between the instrument set I and scan segment set S. The construction of the *connectivity matrix* is shown in Fig. 7.

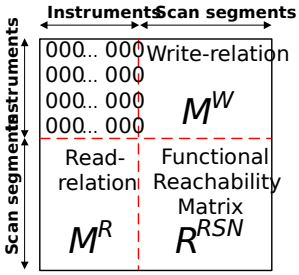

Fig. 7. Connectivity matrix construction

The *functional reachability matrix* R^{RSN} is placed in the lower right part of the *connectivity matrix*, whereas the *read-* and *write*-relations M_r and M_w are transformed into a matrix representation and placed respectively in the lower left and upper right part of *connectivity matrix*. The upper left part of the *connectivity matrix* is filled with the zero values, since at this step the paths between the instruments through the RSN are not computed yet. $R^{connect}$ describes the adjacency of instruments to scan segments, but not yet the reachability from and to instruments. Again, the transitive closure provides a *global reachability matrix* $R^{global} \in \mathbb{B}^{m \times m}$:

$$R^{global} := TCl(R^{connect}) \tag{12}$$

The computation of (12) will stop after computing $(R^{connect})^3$:

- After computing $(R^{connect})^2$, so called 'bridge'-connections between instruments and scan segments or vice versa are inferred.
- After adding $(R^{connect})^3$, all the paths between instruments through the RSN are generated.

The complete data flow in the circuit after RSN integration has to consider transfer in the initial system described by R_c^{init}, transfer in the RSN described by R^{global}, and any combination of them over hybrid paths [14]. This is achieved by augmenting R^{global} (12) with the information about possible data paths in the circuit R_c^{init} (3).

$$r_{k,l}^{hybrid} := \begin{cases} r_{k,l}^{c} \vee r_{k,l}^{global}, & \text{if } (k < |I|) \wedge (l < |I|) \\ r_{k,l}^{global}, & \text{otherwise,} \end{cases} \tag{13}$$

where ($r_{k,l}^{c} \in R_c^{init}$) and ($r_{k,l}^{global} \in R^{global}$) are the corresponding elements in the *compacted restricted reachability matrix* and in the *global reachability matrix*.

The transitive closure of R^{hybrid} describes all connected segments and instruments via an initial data path of the RSN, and the connectivity of instruments is described by the matrix $R^{total} \in \mathbb{B}^{|I| \times |I|}$:

$$R^{total} \subset TCl(R^{hybrid}) \tag{14}$$

The compliance of the RSN with security requirements is checked by computing the syndrome matrix:

$$SD := R_c^{init} - R^{total}, \tag{15}$$

and any $sd_{k,l} < 0, k = l$, denotes a new data flow not in the original circuit and causes a security violation warning.

B. Example

1) Security violation warning: In this section the security compliance analysis flow is described, considering the example represented in Fig. 2. The graph representation of the example system is shown in Fig. 8. The upper part shows the connections in the initial circuit, the lower part represents the connections within the RSN. The dashed arrows represent the connections between instruments I and scan segments S.

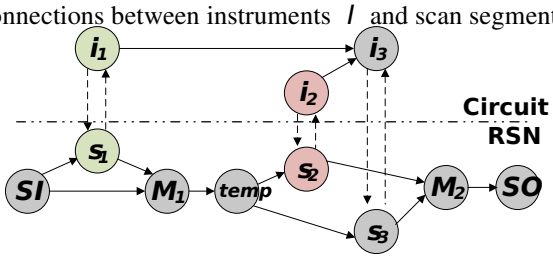

Fig. 8. Graph representation of ex.1

The structure of the initial circuit is simplified and represents only the allowed connections between the instruments. These connections are represented in the *reachability matrix* R_c^{init} (Fig. 9.b), which is computed from the *adjacency matrix* A^{init} (Fig. 9.a). The order of the rows and the columns is i_1, i_2, i_3. The *reachability matrix* R_c^{init} show that data transfer from i_1 to i_3 and from i_2 to i_3 are possible and not restricted by the system designer. The elements on the main diagonal represent the self-reachability of the instruments. Other combinations of instruments to perform data transfer are prohibited.

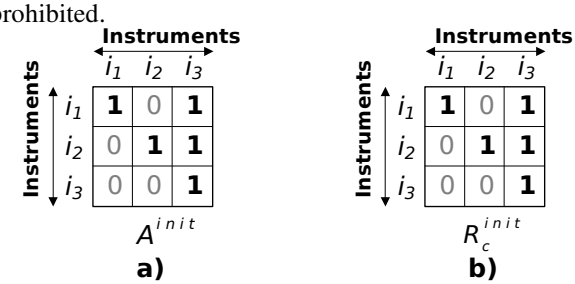

Fig. 9. a) Adjacency matrix of the initial circuit b) Reachability matrix of the initial circuit

The *structural reachability matrix* R_{seg}^{RSN} is computed (Fig. 10.b) from the *adjacency matrix* A^{RSN} of the RSN (Fig. 10.a). The *adjacency matrix* contains information about the connections between all scan primitives including the scan multiplexers M_1 and M_2. In R_{seg}^{RSN} only the reachability of the scan segments is considered. The order of the rows and the columns in R_{seg}^{RSN} is $s_1, temp, s_2, s_3$. At this step,

only structural connections, considering the direction of data propagation between the scan segments, are taken into account.

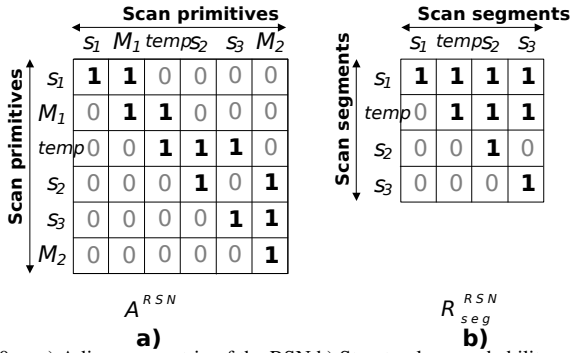

Fig. 10. a) Adjacency matrix of the RSN b) Structural reachability matrix of the RSN (ex.1)

The *essential select conditions* (*ESC*) for all scan segments are computed. To include s_1 into the *ASP* the first branch of the first multiplexer M_1 must be selected:

$$f_1 := (M_1 = 0); \bar{f}_1 := (control = 0) \quad (16)$$

At the same time, the second branch of the second multiplexer M_2 must be selected to include s_2 into the *ASP*:

$$f_2 := (M_2 = 0); \bar{f}_2 := (control = 1) \quad (17)$$

The scan segment $temp$ is included into the *ASP* by any assignment of the logic control signals. The *ASP reachability matrix* R^{RSN}_{path} is represented in Fig. 11.a.

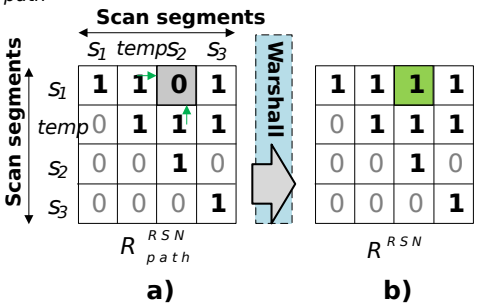

Fig. 11. a) ASP reachability matrix, b) Functional reachability matrix of the RSN (ex.1)

The data transfer from s_1 to s_2 is not functionally possible within a single *ASP*, since the *ESC*s for s_1 and s_2 are contradicting and a satisfying assignment for the values of the logic signals cannot be found. Consequently, the corresponding element of the matrix equals to 0.

The *functional reachability matrix* R^{RSN} is shown in Fig. 11.b and demonstrates that the data transfer from s_1 to s_2 is possible through a reconfiguration of the *ASP*. Since there exists an assignment for the values of the logic signals to propagate data from s_1 to $temp$ ($control = 0$) and another assignment to shift data from $temp$ to s_2 ($control = 1$), data from s_1 can be propagated to s_2 and the corresponding matrix element is equal to 1.

The intermediate *connectivity matrix* $R^{connect}$ for this example is shown in Fig. 12. The rows and columns are represented in the following order: i_1, i_2, i_3, s_1, $temp$, s_2, s_3. The *connectivity matrix* is composed using the *functional reachability matrix* R^{RSN} and *read*- and *write*-relations between instrument set I and scan segment set S as described in Section V-A.

The *global reachability matrix* R_{global} (Fig. 12.b) represents the possible paths between the instruments through the RSN only. The data propagation from i_1 to i_2 is possible through the RSN. However, not all possible paths have been considered at this step. E.g. the path from i_2 to i_3 is not possible only considering the paths inside the RSN (R_{global}). However, such path already exists in the initial circuit (R^{init}_c) and must be considered by the reachability analysis.

The *hybrid reachability matrix* R^{hybrid} augments the reachability properties of the initial circuit with the dependencies, introduced through the RSN integration and is represented in Fig. 12.c. All possible paths between the instruments and scan segments in the augmented circuit are represented in Fig. 12.d by the transitive closure of R^{hybrid}. The dependencies between the instruments in the original circuit R^{init}_c are compared with the ones in the augmented system $R^{total} \subset TCl(R^{hybrid})$ and the *syndrome matrix* is computed as in Fig. 13.

In the provided example a warning is generated due to unwanted data transfer between instruments i_1 and i_2, introduced through the RSN integration and the corresponding element of the *syndrome matrix* equals to -1.

2) Security compliant RSN: In Fig. 14 an example RSN integration is shown, which does not violate the security properties of the initial circuit. Since the same initial circuit is used, the reachability matrix R^{init}_c of the initial circuit is the same as in Section V-B1.

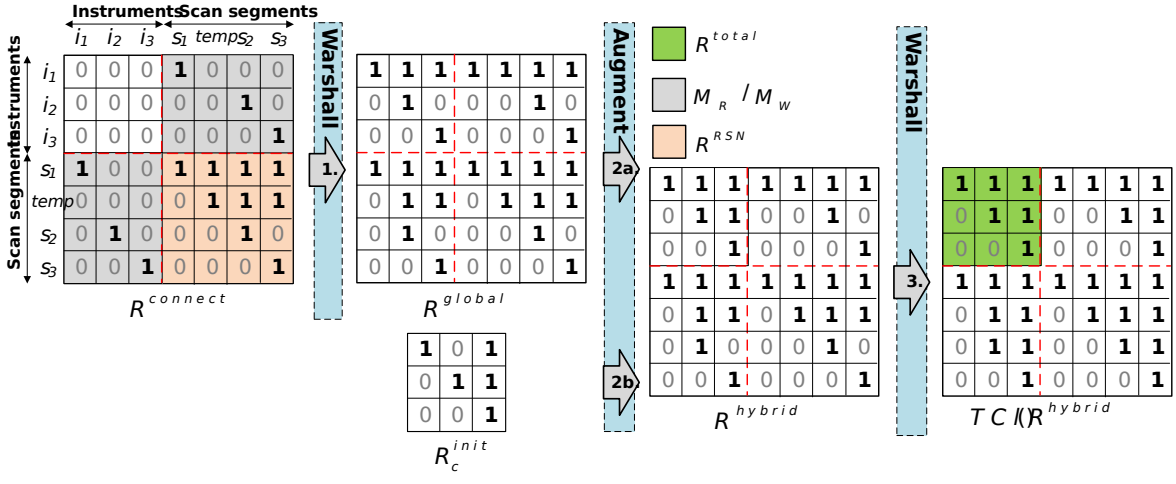

Fig. 12. Matrix-based verification flow for ex.1

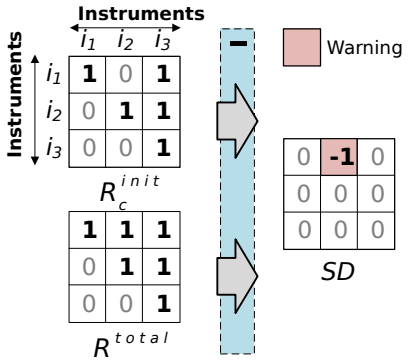

Fig. 13. Syndrome matrix computation for ex.1

The graph representation of the augmented system is constructed as in Section V-B1.

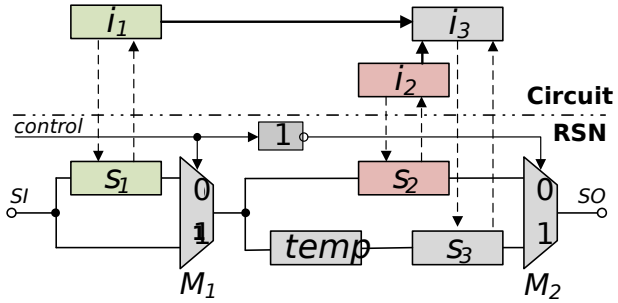

Fig. 14. No additional dependencies introduced

The structural reachability matrix R_{seg}^{RSN} is shown in Fig. 15.a. The *ESC*s of the scan segments s_1 and s_2 remain the same and are computed as in equations (16), (17). The *ASP reachability matrix* is represented in Fig. 15.b and shows that direct data transfer from s_1 to s_2 is still not possible within one scan configuration. Compared to the example, presented in V-B1, no intermediate scan segment between s_1 and s_2 exists, which could be used to transfer data between the scan segments, even if an ASP, including both s_1 and s_2, does not exist. The *functional reachability matrix* (Fig. 15.b) shows that data propagation from s_1 to s_2 is generally impossible, also if multiple reconfigurations are applied to the *ASP*.

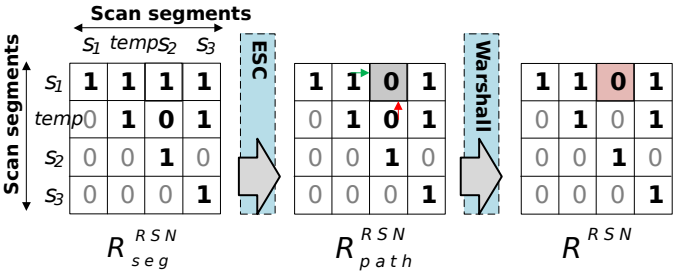

Fig. 15. Reachability matrix computation for ex.2

Fig. 16 represents the *global reachability matrix* R^{global} (a) and the transitive closure of the *hybrid reachability matrix* (b) $TC(R^{hybrid})$. The presented example shows that no additional path between instruments i_1 and i_2 is introduced after the RSN integration, not only considering the *pure paths* inside the RSN (R^{global}), but also through the *hybrid paths* ($TC(R^{hybrid})$).

Since the possible connections between the instruments in the initial system R_c^{init} coincide with the connections in the augmented system after the RSN integration R^{total}, the

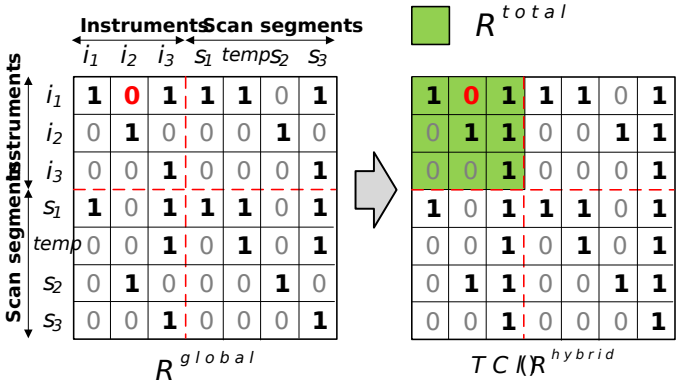

Fig. 16. a) The global reachability matrix, b) The transitive closure of the hybrid reachability matrix for ex.2

syndrome matrix does not contain any negative numbers. No warning is generated, since the analyzed RSN is compliant with the initial circuit.

VI. Evaluation

A. Experimental Setup

The proposed verification method is evaluated on a number of reconfigurable scan networks from the BASTION benchmark set [25]. The proposed algorithm is implemented in the *eda1687* framework as introduced in [18]. All experiments are conducted on a single core, Intel Core i7-5600U CPU at 2.60GHz with 8 GB of main memory.

The benchmarks from the BASTION benchmark set provide access to boundary and internal scan chains and have hierarchical structure. In Table I characteristics of the considered benchmarks are presented. For each benchmark the information about the number of scan multiplexers (Column 2) and SIBs (Column 3) is given. The fourth and the fifth column define the quantity of scan segments and scan cells respectively. The number of hierarchy levels is given in the sixth column.

TABLE I. Characteristics of benchmarks[25]

Design	Benchmark characteristics				
	#muxes	#sibs	#segments	#scan cells	#level
BasicSCB	10	-	21	176	4
Mingle	13	10	270	22	3
TreeFlat	24	12	24	101	2
TreeUnbalanced	28	28	63	41,887	11
TreeBalanced	46	43	90	5,581	7
TreeFlat_Ex	60	57	123	5,194	5
q12710	25	25	47	26,183	2
a586710	47	-	79	41,682	3
p34392	142	-	245	23,261	3
t512505	160	-	288	77,006	2
p22810	283	283	537	30,111	3
p93791	653	-	1 241	98,637	3
N17D3	8	7	11	447	3
N32D6	10	13	24	9,6135	4
N73D14	17	29	56	218,823	11
N132D4	40	39	92	2,912	5

The different RSNs (Table I) were inserted into the same circuit description defining the matrix R_c^{init}, in order to obtain comparable results. The connectivity was not randomly generated, but taken from the ISCAS'89 s298 benchmark, to avoid any possible bias in the experiments. The instruments in the circuit have been modeled by flip flops. The

connections between the instruments have been extracted from the initial circuit description. The corresponding reachability matrix has been computed as in Section III. Additional explicit permissions on data transfers between instruments, have been generated randomly and constitute 5% of theoretically possible connections between instruments.

B. Functional reachability

For each RSN a series of experiments has been conducted and the *functional reachability fraction* has been computed.

Definition 7: The *functional reachability fraction* is a fraction of functionally reachable scan segment pairs compared to the total number of structurally reachable scan segment pairs:

$$frac_{func} = \frac{\#connect_{func}}{\#connect_{struct}} \tag{18}$$

A certain fraction (from 0% to 100% in 10% steps) of scan multiplexers from a total number of scan multiplexers was divided into *control groups* of size n. All scan multiplexers in one *control group* were controlled by the same *control*-signal and are called *dependent* multiplexers. The *control groups* including 2 and 10 scan multiplexers have been investigated. For each *control group* the first scan multiplexer is chosen randomly. Since the scan multiplexers tend to share the logic control signal with neighbors more often than with the scan multiplexers located in the different parts of the circuit, another $n-1$ *dependent* scan multiplexers in the group are chosen nearby the first scan multiplexer. Distance D defined the number of scan primitives between two neighboring multiplexers. So if the second multiplexer in a *control group* has the distance D from the first multiplexer, the j-th multiplexer has the distance of $(j-1)\cdot D$ from the first one. The distance D equals 1 in the experiment.

In Fig. 17 the *functional reachability fraction* has been calculated for the *p34392* benchmark.

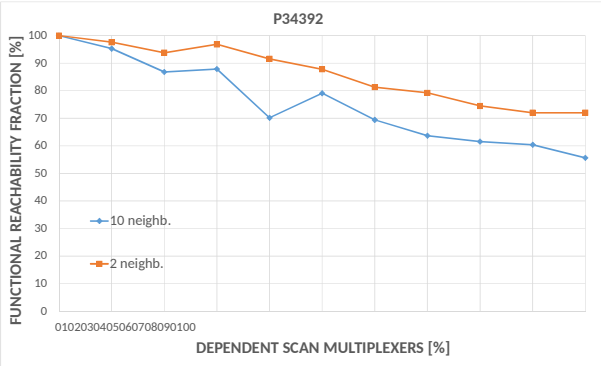

Fig. 17. Functional reachability fraction for p34392 benchmark

Increasing the number of *dependent* scan multiplexers from 0% to 100% causes that the *functional reachability fraction* is decreasing for all group sizes. With the increase of a group size n more functional restrictions are added to the benchmark and less connections between the scan segments remain functionally possible. Consequently, the *functional reachability fraction* for the group size of 10 is lower than the one for pairwise *dependent* scan multiplexers ($n=2$).

Since the difference between the number of possible connections in the benchmark reaches up to three orders of magnitude, the *weighted arithmetic mean* was used to compute the *functional reachability fraction*.

$$\overline{frac}_{func} = \frac{\sum_{j=1}^{n} w_j \cdot (frac_{func})_j}{\sum_{j=1}^{n} w_j}, \tag{19}$$

where $(frac_{func})_j$ is the *functional reachability fraction* of the benchmark j, w_j is the weight of the benchmark j, depending on the number of structurally possible connections.

The *functional reachability fraction* over all used benchmarks has been computed and is shown in Fig. 18. For all considered benchmarks in average with the increase of a *control group* size and of a *dependent* scan multiplexers fraction, the *functional reachability fraction* decreases, since less connections between scan segments remain functionally reachable.

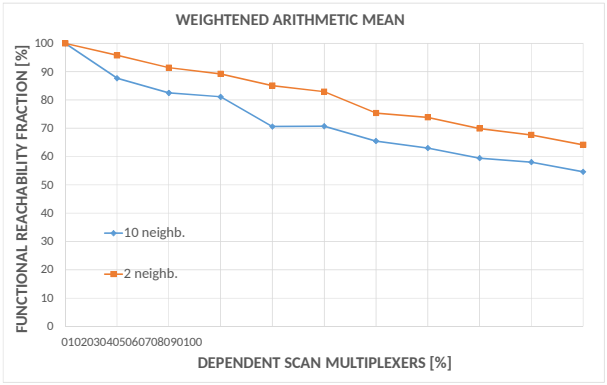

Fig. 18. Functional reachability fraction (weighted arithmetic mean)

C. Security compliance analysis

The presented approach has been used to analyze the reachability properties of the integrated RSN and to verify, whether the RSN integration is violating the security properties of the initial design. It was assumed that 30% of scan multiplexers were divided into the *control groups* of size 2. Each experiment was run 20 times to investigate different randomized samples of scan multiplexer pairs, which are controlled by the same control signal. After integrating the RSN to the circuit it has been verified, whether the RSN introduced additional data paths between the instruments compared to the initial design, as described in Section V.

Table II shows the results for security compliance analysis for benchmarks from the BASTION set, integrated into the initial circuit.

For each benchmark the number of structurally reachable scan segment pairs (column 2) has been computed. The average number of functionally reachable pairs of scan segments, introduced by single active paths (column 3) and by retargeting (column 4) have been calculated. The average number of scan segment pairs violating the security properties in the initial circuit is presented in Column 5.

Using the presented approach an accurate estimation of reachability properties of all considered benchmarks is provided. Since the set of structurally reachable scan segment pairs is an over-approximation of the functionally reachable pairs, the number of structural connections (Column 2) is always greater than or equal to the number of the functionally possible connections (Column 4). For all considered benchmarks the number of functionally reachable scan segment pairs

TABLE II. Reachability matrix computation by 30% dependent scan multiplexers

Design	Matrix characteristics				
	#struct	# ASP	#func	#viol	t[m:s]
BasicSCB	181	148	156	45	00:10
Mingle	220	196	199	70	00:15
TreeFlat	300	268	297	143	01:00
TreeUnbalanced	2,016	1,244	1,285	678	01:55
TreeBalanced	4,272	2,626	2,879	1,459	01:30
TreeFlat_Ex	7,869	7,003	7,052	4,015	02:35
q12710	1,275	1,207	1,245	563	00:35
a586710	1,430	955	1,187	331	00:50
p34392	15,937	14,898	15,432	8,967	05:30
t512505	41,328	36,675	39,869	13,778	20:15
p22810	137,550	132,959	134,424	56,349	30:45
p93791	721,269	622,759	627,570	264,221	55:40
N17D3	895	752	770	235	00:20
N32D6	2,268	2,051	2,164	1,780	01:55
N73D14	9,299	6,896	6,967	3,865	03:15
N132D4	31,586	30,557	30,776	13,513	06:50

in average exceeds number of scan segments pairs, reachable within single active scan path (Column 3). E.g. for the *t512505* benchmark in average 3194 additional paths are introduced by retargeting and cannot be investigated by simple scan path analysis. This emphasizes the use of an accurate analysis method, which avoids false positives and false negatives by security compliance verification.

The runtime of the presented algorithm (Column 6) is highly affected by the number of multiplexers in the benchmark. The worst runtime (about one hour) was achieved for *p93791* benchmark with the highest number of multiplexers. However, the runtime for most benchmarks is around 2-5 minutes.

This experiment illustrates that the proposed approach is suitable for the analysis if an integrated RSN will introduce any security violations.

VII. Conclusion

Even though test infrastructure poses a security threat, it cannot be disconnected from the circuit after the manufacturing, since the access to on-chip infrastructure should be remained for the whole system lifecycle for monitoring and maintenance.

This paper proposes a method to analyze and detect all side-channels opened by RSN infrastructure integration. Complex dependencies in the functional part of the system are analyzed to formulate the restrictions to RSNs. Restrictions to the RSN are verified and the set of security violations warnings is generated. The applicability of proposed method is evaluated on a wide set of benchmarks. The proposed method adds an acceptable runtime (up to one hour for the biggest benchmark) to the security verification flow even for large benchmarks.

Acknowledgments

This work was financed by Baden-W ürttemberg Stiftung (IKT-Sicherheit, SHIVA).

Bibliography

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

[2] "IEEE Standard for Access and Control of Instrumentation Embedded within a Semiconductor Device," *IEEE Std 1687-2014*, pp. 1–283, Dec. 2014.

[3] "IEEE Standard for Test Access Port and Boundary-Scan Architecture," *IEEE Std 1149.1-2013 (Revision of IEEE Std 1149.1-2001)*, pp. 1–444, May 2013.

[4] F. G. Zadegan, E. Larsson, A. Jutman, S. Devadze, and R. Krenz-Baath, "Design, Verification, and Application of IEEE 1687," in *Proc. IEEE Asian Test Symp. (ATS)*, Nov. 2014, pp. 93–100.

[5] J. D. Rolt, A. Das, G. D. Natale, M. Flottes, B. Rouzeyre, and I. Verbauwhede, "Test Versus Security: Past and Present," *IEEE Trans. on Emerging Topics in Computing*, vol. 2, no. 1, pp. 50–62, Mar. 2014.

[6] B. Yang, K. Wu, and R. Karri, "Secure Scan: A Design-for-Test Architecture for Crypto Chips," *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems (TCAD)*, vol. 25, no. 10, pp. 2287–2293, Oct. 2006.

[7] 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)*, Apr. 2015, pp. 1–4.

[8] J. Dworak, A. Crouch, J. Potter, A. Zygmontowicz, and M. Thornton, "Don't forget to lock your SIB: hiding instruments using P1687," in *Proc. IEEE Int'l Test Conf. (ITC)*, Sept. 2013, pp. 1–10.

[9] H. Liu and V. D. Agrawal, "Securing IEEE 1687-2014 Standard Instrumentation Access by LFSR Key," in *Proc. IEEE Asian Test Symp. (ATS)*, Nov. 2015, pp. 91–96.

[10] A. Zygmontowicz, J. Dworak, A. Crouch, and J. Potter, "Making it harder to unlock an LSIB: Honeytraps and misdirection in a P1687 network," in *Proc. Conf. on Design, Automation Test in Europe (DATE)*, Mar. 2014, pp. 1–6.

[11] S. Kan and J. Dworak, "IJTAG Integrity Checking with Chained Hashing," in *Proc. IEEE Int'l Test Conf. (ITC)*, Oct. 2018, pp. 1–10.

[12] M. A. Kochte, R. Baranowski, and H.-J. Wunderlich, "Trustworthy Reconfigurable Access to On-Chip Infrastructure," in *Proc. IEEE Int'l Test Conf. in Asia (ITC-Asia)*, Sep. 2017.

[13] R. Elnaggar, R. Karri, and K. Chakrabarty, "Securing IJTAG against data-integrity attacks," in *Proc. IEEE VLSI Test Symp. (VTS)*, Apr. 2018, pp. 1–6.

[14] P. Raiola, B. Thiemann, J. Burchard, A. Atteya, N. Lylina, H.-J. Wunderlich, B. Becker, and M. Sauer, "On Secure Data Flow in Reconfigurable Scan Networks," in *Proc. Conf. on Design, Automation Test in Europe (DATE)*, Mar. 2019, pp. 1–6.

[15] A. Atteya, M. A. Kochte, M. Sauer, P. Raiola, B. Becker, and H.-J. Wunderlich, "Online Prevention of Security Violations in Reconfigurable Scan Networks," in *Proc. IEEE European Test Symp. (ETS)*, May 2018, pp. 1–6.

[16] R. Baranowski, M. A. Kochte, and H.-J. Wunderlich, "Access Port Protection for Reconfigurable Scan Networks," *Journal of Electronic Testing: Theory and Applications (JETTA)*, vol. 30, no. 6, pp. 711–723, 2014.

[17] P. Raiola, M. A. Kochte, A. Atteya, L. R. Gomez, H.-J. Wunderlich, B. Becker, and M. Sauer, "Detecting and Resolving Security Violations in Reconfigurable Scan Networks," in *Proc. IEEE Int'l Symp. on On-Line Testing And Robust System Design (IOLTS)*, Jul. 2018, pp. 91–96.

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

[19] K. Nakamura, K. Takagi, S. Kimura, and K. Watanabe, "Waiting false path analysis of sequential logic circuits for performance optimization," in *Proc. IEEE/ACM Int'l Conf. on Computer-Aided Design. Digest of Technical Papers (ICCAD)*, Nov 1998, pp. 392–395.

[20] Z. Hanna and V. M. Purri. (2013, Apr.) Verifying Security Aspects of SoC Designs with Jasper App. [Online]. Available: https://www.edn.com/

[21] M. Soeken, P. Raiola, B. Sterin, B. Becker, G. De Micheli, and M. Sauer, *Proc. 12th Int'l Haifa Verification Conference (HVC)*. Springer, 2016, ch. SAT-Based Combinational and Sequential Dependency Computation, pp. 1–17.

[22] A. Mondal, P. P. Chakrabarti, and C. R. Mandal, "A new approach to timing analysis using event propagation and temporal logic," in *Proc. Design, Automation and Test in Europe Conf. and Exhibition (DATE)*, vol. 2, Feb. 2004, pp. 1198–1203 Vol.2.

[23] S. Warshall, "A Theorem on Boolean Matrices," *Journal of the ACM (JACM)*, vol. 9, no. 1, pp. 11–12, Jan. 1962.

[24] F. G. Zadegan, R. Krenz-Baath, and E. Larsson, "Upper-bound computation for optimal retargeting in IEEE 1687 networks," in *Proc. IEEE Int'l Test Conf. (ITC)*, Nov. 2016, pp. 1–10.

[25] A. Tsertov, A. Jutman, S. Devadze, M. S. Reorda, E. Larsson, F. G. Zadegan, R. Cantoro, M. Montazeri, and R. Krenz-Baath, "A suite of IEEE 1687 benchmark networks," in *Proc. IEEE Int'l Test Conf. (ITC)*, Nov. 2016, pp. 1–10.