Modeling, Verification and Pattern Generation for Reconfigurable Scan Networks

Baranowski, Rafal; Kochte, Michael A.; Wunderlich, Hans-Joachim

Proceedings of the IEEE International Test Conference (ITC'12) Anaheim, California, USA, 5-8 November 2012

doi: http://dx.doi.org/10.1109/TEST.2012.6401555

Abstract: Reconfigurable scan architectures allow flexible integration and efficient access to infrastructure in SoCs, e.g. for test, diagnosis, repair or debug. Such scan networks are often hierarchical and have complex structural and functional dependencies. For instance, the IEEE P1687 proposal, known as IJTAG, allows integration of multiplexed scan networks with arbitrary internal control signals. Common approaches for scan verification based on static structural analysis and functional simulation are not sufficient to ensure correct operation of these types of architectures. Hierarchy and flexibility may result in complex or even contradicting configuration requirements to access single elements. Sequential logic justification is therefore mandatory both to verify the validity of a scan network, and to generate the required access sequences. This work presents a formal method for verification of reconfigurable scan architectures, as well as pattern retargeting, i.e. generation of required scan-in data. The method is based on a formal model of structural and functional dependencies. Network verification and pattern retargeting is mapped to a Boolean satisfiability problem, which enables the use of efficient SAT solvers to exhaustively explore the search space of valid scan configurations.

Preprint

Modeling, Verification and Pattern Generation for Reconfigurable Scan Networks

Rafal Baranowski, Michael A. Kochte, Hans-Joachim Wunderlich

ITI, University of Stuttgart, Pfaffenwaldring 47, D-70569, Stuttgart, Germany
Email: {baranowski, kochte}@iti.uni-stuttgart.de, wu@informatik.uni-stuttgart.de

***Abstract*—Reconfigurable scan architectures allow flexible integration and efficient access to infrastructure in SoCs, e.g. for test, diagnosis, repair or debug. Such scan networks are often hierarchical and have complex structural and functional dependencies. For instance, the IEEE P1687 proposal, known as IJTAG, allows integration of multiplexed scan networks with arbitrary internal control signals.**

Common approaches for scan verification based on static structural analysis and functional simulation are not sufficient to ensure correct operation of these types of architectures. Hierarchy and flexibility may result in complex or even contradicting configuration requirements to access single elements. Sequential logic justification is therefore mandatory both to verify the validity of a scan network, and to generate the required access sequences.

This work presents a formal method for verification of reconfigurable scan architectures, as well as pattern retargeting, i.e. generation of required scan-in data. The method is based on a formal model of structural and functional dependencies. Network verification and pattern retargeting is mapped to a Boolean satisfiability problem, which enables the use of efficient SAT solvers to exhaustively explore the search space of valid scan configurations.

***Keywords*-Reconfigurable scan network, Pattern generation, Pattern retargeting, DFT, IJTAG, P1687**

I. Introduction

Reconfigurable scan architectures are a cost-efficient and scalable mechanism to allow access to on-chip infrastructure like monitoring, test, debug, diagnosis or reliability. In the field of testing, different types of configurable scan architectures have been proposed for test time reduction [1], [2], [3], for built-in self test [4], [5], for test compression and power reduction [6], for improved core isolation [7], and efficient test access mechanisms for concurrent SoC testing [8], [9]. Configurable scan access can also serve as a light-weight access to debug infrastructure [10], [11].

The standardization of scan access allows plug and play reuse of intellectual property (IP). JTAG (IEEE Std. 1149.1) allows an instruction register to select a number of test-data registers. IEEE Std. 1500 test wrappers provide a configurable serial and parallel interface to a core's scan chains. The forthcoming IEEE Std. P1687, also known as IJTAG, allows highly flexible reconfigurable scan architectures with distributed and hierarchical configuration in various topologies [12].

Complex reconfigurable scan architectures may include fan-outs and reconvergences of scan paths via multiplexers, and arbitrary Boolean functions for internal control signals. With IP reuse, such scan networks may be composed of third party modules, the behavior of which may not be fully understood or not well defined. As a consequence, integration issues may occur, such as exclusive or limited access to certain scan registers. Certain configurations may be illegal or contradictory and require an exhaustive search to find a valid access sequence. The scan architecture may be hierarchical and control signals for scan registers may depend on other scan registers in the same or different hierarchy level.

An example is given in figure 1, where the access to the second module is controlled by the registers of the first module. Clearly, there exists no assignment to registers a and b such that the second module is part of the active scan path. Combinational ATPG can be used to prove module 2 inaccessible because of the combinational nature of this conflict. However, conflicts may very well be sequential and hard to identify due to the high sequential depth of reconfigurable scan networks. Deep sequential search space exploration may also be required to find an access sequence to irredundant modules.

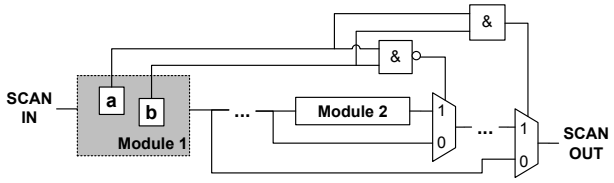

Figure 1. Reconfigurable scan network with conflicting access conditions

For non-configurable scan architectures, simple electrical and logical design rules are sufficient to guarantee correct operation. Electrical requirements are verified with static timing analysis [13], and correct operation is validated by coverage-driven simulation [14]. The integrity of such scan architectures can be checked by expert systems [15], or structural traversal of the network to find multiple drivers, broken chains, or loop-backs [16].

Observability and controllability of scan elements in non-configurable architectures require that a primary input sensitizing condition (called scan state) exists, such that the

scan network can function as a shift register [17]. For reconfigurable scan networks, this is not anymore sufficient, since the sensitizing condition may relate to the internal state of the network. Thus, formal logic reasoning based on a formal model of the scan architecture including internal and external constraints is required to prove integrity of configurable networks, and to generate the required access sequences.

Early research in reconfigurable scan architectures includes performance evaluation [18] and optimal construction of scan hierarchies [19]. These approaches are restricted to architectures, where reconfigurability is limited to hierarchical gateways.

To our knowledge, this paper presents the first method to represent reconfigurable scan structures including all required configuration and control dependencies in a formal way. The network structure is modeled as a directed graph, while all internal combinational and sequential constraints are modeled as logic predicates of graph nodes. The node predicates are transformed into a set of clauses for a Boolean satisfiability (SAT) problem such that only valid scan accesses to elements in the network satisfy the problem instance. A SAT solver is used to exhaustively search the problem space for valid accesses and access sequences, or prove that no such accesses exist. Consequently it is more robust than post-processing, greedy heuristics or algorithms based on static structural exploration without backtracking.

The proposed method is applicable to both serial and parallel scan structures. Hierarchical and exclusive access, as well as arbitrary Boolean functions for control signals are supported.

The following section introduces the used terminology. Sections III and IV describe the modeling of the network and scan accesses. Section V discusses the application to verification and scan pattern generation, followed by experimental results in section VI.

II. Terminology

The existing standards for reconfigurable scan architectures put various constraints on network topology and access mechanisms. For instance, the IEEE Std. 1500 defines the precise architecture and functioning of a scan network, while the IEEE Std. P1687 proposal allows nearly arbitrary topologies and user-defined access mechanisms. In this paper, we follow a general definition of a reconfigurable scan network that covers the existing standards to a large extent, including the forthcoming P1687. In general, reconfigurable scan networks can be decomposed into basic building components, such as scan registers, multiplexers, or combinational logic blocks.

A *scan segment* is a shift register composed of one or more single-bit *scan registers* sharing a set of configuration signals. Data is shifted from the segment's scan-input, through its register bits, to the scan-out of the segment.

A scan segment has from one up to three control signals:

- Select (*sel*): a mandatory control signal that specifies if the access to the scan segment is enabled for capture, shift, and update operation.
- Update disable (*updis*): an optional signal that invalidates the update operation on the scan segment, regardless of the *sel* signal.
- Capture disable (*capdis*): an optional signal that invalidates the capture operation, regardless of the *sel* signal.

A *scan network* is a netlist consisting of scan segments, multiplexers, their control signals and related combinational logic. The control signal of a scan multiplexer is called *address* and specifies the selected scan input. The state of control signals is determined by external control inputs, state of arbitrary scan registers, and Boolean functions thereof. A scan network has one or more *primary scan inputs* and one or more *primary scan outputs*.

Scan segments consist of scan cells with two latches: a *scan/capture* latch and an *update* latch, as in IEEE Std. 1149.1 test data registers (TDR). The first latch is part of the scan chain: It captures data during *capture* operation, and its content is shifted during *shift* operation. The update latch is loaded from the scan latch during *update* operation and remains stable during *shift* operation. Such scan cells are used to drive internal control signals to assure signal stability. Scan segments that do not drive any control signals, e.g. internal scan chains, do not need to include the update latches.

Two scan segments are *directly connected* if their scan-out and scan-in ports are connected either by a net or through a multiplexer. A *scan path* is a non-circular sequence of directly connected scan segments starting at a primary scan-in port 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 asserted and all on-paths multiplexers address the input that belongs to the active scan path.

A *scan configuration* is the state of all sequential elements of a scan network. A *default* scan configuration is the state after the network is reset, or powered up if no reset exists. It is assumed that in the default scan configuration all sequential elements are in a known state (either a '0' or a '1').

A scan configuration is *valid* if and only if: (i) an active path is well formed (all on-path scan segments are selected and on-path multiplexers appropriately addressed) and (ii) scan segments that do not belong to the active scan path are deselected.

The basic access to the scan network is an atomic (inseparable) operation that consists of three phases: *capture, shift,*

and update (CSU). During capture, the registers on the active scan path may latch new data. This data is shifted out during the shift phase, while new scan data is shifted in. Finally, the shifted-in data is latched in the scan registers on the active scan path.

A read or write access to a scan register in the network requires that the accessed register is part of an active scan path. In a reconfigurable scan network, a CSU operation may change the active scan path and affect the next CSU.

A *scan access* is a sequence of CSU operations required to reconfigure the scan network and access the target registers. The *length* of the scan access is the number of constituent CSU operations.

For the sake of simplicity, we assume that:

- Except for multiplexers, a scan path does not contain any combinational elements.
- The active scan path does not branch, i.e. if a scan segment is enabled, only one of its direct successors is enabled.

Embedded test structures with known access mechanisms, such as test compression or compaction architectures, are integrated here as black boxes. Such test-specific subnetworks are represented by scan segments. The proposed modeling method can be used to find an access sequence in the RSN such that the subnetwork becomes part of the active scan path and can be accessed.

III. Modeling of the Scan Network

The following sections describe the modeling of a reconfigurable scan network as a graph and a set of node predicates which reflect structural and path activation constraints. The predicates are transformed into a Boolean satisfiability problem.

A. Graph Representation

The structure of a scan network is modeled with a directed graph $G = (V, E)$. The set of vertices V is partitioned into segment nodes V_S, primary nodes V_P, and auxiliary nodes V_A. The set V_P represents primary scan-in and scan-out ports, V_S represents scan segments, and V_A is used to model multiplexers and fanout branches. The set of edges $E \subseteq V \times V$ represents connections between scan network elements such that an edge $e = (v_1, v_2) \in E$ if and only if the two elements corresponding to $v_1, v_2 \in V$ are directly connected.

A *scan path* in G is defined as a directed path from a primary scan-in node $v_{in} \in V_P$, through zero or more directly connected nodes $v_s \in (V_S \cup V_A)$, to a primary scan-out node $v_{out} \in V_P$.

A graph node $v \in V_S$ is defined *active* if and only if the select signal of the corresponding scan segment is asserted: $sel(v) = 1$. Otherwise the node is *inactive*, $sel(v) = 0$.

An *active scan path* in G is defined as a scan path in G that contains only active nodes.

Fig. 2 presents an example of a scan network with four scan segments, a single scan fan-out, and a scan multiplexer (a), and its corresponding graph representation (b).

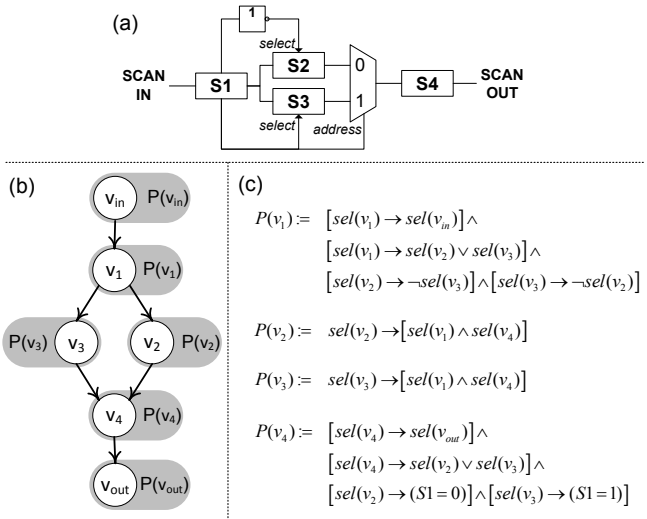

Figure 2. Example of (a) a scan network (b) its corresponding graph and (c) node predicates

In order to simplify the inference of node predicates (section III-B), auxiliary nodes are added for multiplexers and fanout branches when:

(a) A fanout branch drives a multiplexer with no intermediate scan segment
(b) A multiplexer constitutes a fanout stem
(c) A multiplexer drives another multiplexer with no intermediate scan segment

Fig. 3 shows an example of all the three cases, together with the corresponding scan graphs augmented with auxiliary nodes.

If a multiplexer or a fanout branch drives a single scan segment, no auxiliary graph node is required.

B. Predicates

The predicate function $P : V \mapsto B$ assigns each node $v \in V$ a Boolean predicate $P(v) \in B$. Predicate $P(v)$ evaluates to true if the local scan configuration for node v is valid as explained below:

The predicate function for a node v with a single predecessor p and a single successor s is given by:

$$P(v) := sel(v) \rightarrow [sel(p) \wedge sel(s)]$$

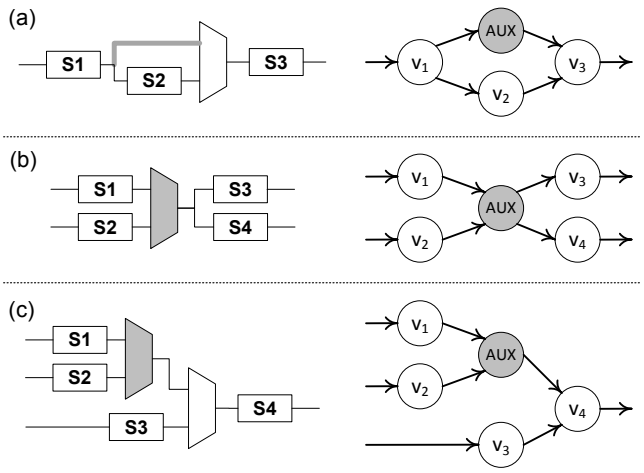

Figure 3. Auxiliary nodes for multiplexers and fanout branches

The predicate requires that if v is active, both its predecessor and its successor are active. In other words, if node v belongs to the active scan path, so do both its predecessor and successor.

For a node v with a single predecessor p and multiple successors $s \in succ(v)$, a valid scan configuration requires that at most one successor of v is active. This is captured by the following predicate:

$$P(v) := [sel(v) \rightarrow sel(p)] \wedge (1) \wedge (2)$$

$$sel(v) \rightarrow \bigvee_{s \in succ(v)} sel(s) \quad (1)$$

$$\forall_{(s_k,s_l) \in succ(v), s_k \neq s_l} : [sel(s_k) \rightarrow \neg sel(s_l)] \quad (2)$$

The predicate assures that in case of a fan-out the active scan path does not branch, i.e. there are not multiple active successors. A fan-out node is active if and only if its predecessor and exactly one of its successors is active (cf. node v_1 in fig. 2).

For a node v with a single successor s and multiple predecessors $p \in pred(v)$, a valid scan configuration requires that at most one predecessor of v is active. If a predecessor of v is active, the *address* of the multiplexer must be correctly set:

$$P(v) := [sel(v) \rightarrow sel(s)] \wedge (3) \wedge (4)$$

$$sel(v) \quad \rightarrow \quad \bigvee_{p \in pred(v)} sel(p) \quad (3)$$

$$\forall_{p \in pred(v)} : [sel(p) \quad \rightarrow \quad address = addr(p)] \quad (4)$$

The predicate assures that in case of a multiplexed scan path the active path is correctly routed. A multiplexed node is active if and only if one of its predecessors is active, its successor is active and the predecessor is correctly selected by the *address* signal of the multiplexer (cf. node v_4 in fig. 2)

In case of a node v with multiple predecessors $p \in pred(v)$ and multiple successors $s \in succ(k)$, the following predicate captures the condition for a valid scan configuration:

$$P(v) := (1) \wedge (2) \wedge (3) \wedge (4)$$

The predicate assures that a scan segment with an input multiplexer and a fan-out is active if and only if exactly one of its preceding and one of its succeeding scan segments are active.

Both the predicates and the scan graph are constructed by parsing a structural model of the scan network: a gate-level netlist, or an abstract representation, for instance in Instrument Connectivity Language (ICL), as proposed by P1687. The scan elements and their dependencies are found by traversing the structural network model from the primary scan inputs.

C. Clausal Representation

The conjunction of the predicates is transformed into a conjunctive normal form (CNF) or a set of clauses. This set is extended with additional clauses that describe the Boolean functions and propagation of control signals. The resulting Boolean formula is satisfiable if and only if an assignment to the variables exists which results in a valid scan configuration. The formula describes structural constraints only and will be referred to as *structural SAT model*. The structural SAT model is the basis of scan access modeling in section IV.

The variables of the structural SAT model represent the state of scan segments, predicates, external control inputs, as well as internal control signals of scan segments (*sel*, *updis*, *capdis*) and scan multiplexers (*address*).

Each 1-bit register j of a scan segment v is assigned a separate variable v_j. This variable is added to the structural SAT instance as a constraint or unit clause that reflects the initial scan configuration: If the initial value of a register is logic '1', its corresponding variable occurs in the instance in a positive form.

We assume that all external control signals and the state of all scan segments are known and can be encoded with two-valued logic. If consideration of unknown values is required, the proposed modeling can be extended to a three-valued logic with unknowns $(0, 1, X)$, as described in [20]. In this case, the structural SAT model must be extended with the following constraints:

- For each node, the value of the corresponding *sel* signal is defined (either 0 or 1).
- For a node with multiple predecessors, the *address* of the multiplexer is defined if the node is active.

The additional constraints assure that no scan data is lost due to unknown values of control signals.

IV. Modeling of Scan Accesses

Both the read and write access to a scan register require that the accessed register is on an active scan path. Given the initial scan configuration, a scan access may require multiple configuration steps to eventually include the targeted element on the active scan path. In each configuration step, a justification of the respective path activation requirements has to be conducted.

The proposed modeling allows to search for justification, required configuration and execution of single as well as multiple CSU operations, or to prove that elements cannot be accessed from a given initial scan configuration.

To account for scan accesses requiring multiple CSU operations, the structural SAT model is unrolled for a finite number of time frames. A transition between two time frames corresponds to one atomic CSU operation. The resulting instance is satisfiable if and only if a sequence of CSU operations exists which implements the scan accesses and ensures that corresponding activation constraints are fulfilled.

The number of required time frames to access any scan register is bounded. The derivation of an upper bound from an arbitrary initial scan configuration is beyond the scope of this paper. In the following, we assume that the user specifies the maximum allowable length of a scan access. If no solution can be found within this bound, the targeted elements are considered unreachable.

A. Temporal Modeling by Unrolling

The structural SAT model encodes all combinational and structural constraints imposed by the scan graph. To correctly model scan accesses with multiple CSU operations, the structural SAT model is unrolled. Distinct variables encode the state of the graph (scan registers and predicates) in the respective time frame.

Fig. 4 shows an example where the model is unrolled k times to reflect k CSU operations. The state of the variables in a single time frame t_i represents the scan configuration after i CSU operations. In the first frame t_0, this may be the default scan configuration or a given initial state.

The state of a scan register may change between two subsequent time frames t_i and t_{i+1} if the update disable signal of the corresponding scan segment is inactive and the scan segment is part of the active scan path in time frame t_i. If the register is disabled or does not belong to the active scan path, its state is stable. For a variable v_j of a scan segment v, this relation between corresponding variables v_j^i, v_j^{i+1} in subsequent time frames is expressed by the implication:

$$(v_j^i \oplus v_j^{i+1}) \rightarrow sel(v^i) \wedge \neg updis(v^i). \quad (5)$$

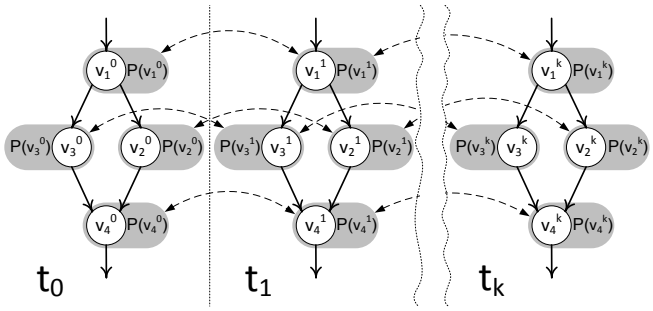

Figure 4. Modeling of k CSU operations by unrolling

I.e. if scan register v_j changes its value from frame t_i to the next frame t_{i+1}, then scan segment v must be enabled and part of an active scan path in frame t_i. Implication (5) is added for each variable in each time frame t, $t_0 \leq t < t_k$.

B. Read and Write Accesses

This section describes the encoding of read and write accesses in the scan network model as constraints to the unrolled model with $k+1$ time frames. The resulting instance is satisfiable if and only if a scan access sequence of length k or shorter exists which accesses the targeted elements as specified.

A read access to register v_j requires that the capture disable signal of the corresponding scan segment v is inactive and the target is on an active scan path at least once during all considered time frames. For $k+1$ frames, this is modeled by the condition:

$$\bigvee_{i=[0,k]} sel(v^i) \wedge \neg capdis(v^i).$$

Similar to the read access, a write access to register v_j with value w requires that the update disable signal of the corresponding scan segment v is inactive, the target is on an active scan path at least once, and in the last time frame the value of v_j equals to w:

$$(v_j^k = w) \wedge \bigvee_{i=[0,k]} sel(v^i) \wedge \neg updis(v^i).$$

V. Application to Verification and Pattern Generation

A. Observability and Controllability Check

To assure that a scan segment can be both read from and written to, it is necessary to prove that it is observable and controllable. A necessary requirement is that there exists a scan path from a primary scan input, through the segment, to a primary scan output. To determine if a structural connection exists, a static connectivity check can be used [13]. For complex scan architectures with arbitrary control signals, the necessary and sufficient requirement is justification of control signals over one or multiple time frames.

We define that a scan segment is *observable and controllable* in scan configuration C if and only if there exists a sequence of CSU operations starting at C, such that the scan segment is part of the active scan path at least once, while the corresponding update and capture disable signals are inactive.

A scan network is defined *valid* if each constituent scan segment is controllable and observable in the default scan configuration (after reset), and the default scan configuration can be restored by a scan access of finite length.

Verification is conducted iteratively over all scan segments with the following constraints: (i) in the first time frame, the network has a default scan configuration, (ii) the target scan segment is accessed at least once, and (iii) after a bounded number of CSU operations, the default scan configuration is restored, except for the target scan segment. As described in section IV-A, the structural SAT model is unrolled until all the constraints are satisfied, i.e. the model is satisfiable, or until a predefined bound for the scan access length is reached. The network is proven valid only if the iterations for all scan segments result in satisfiable SAT instances. Otherwise, the network is considered not valid.

B. Scan Pattern Generation

An access to a scan segment may require several CSU operations to put the target scan segment on the active scan path. The proposed modeling can be used to generate the scan-in data sequences required to access the target elements. The user specification of an access (the set of target registers and data) is encoded in clauses as described in section IV-B.

In IEEE P1687 proposal, the process of computing the required scan-in sequence is called *pattern retargeting*. P1687 proposes to specify the user constraints in Procedural Description Language (PDL), which can be automatically translated into clauses.

The SAT instance is iteratively unrolled, as described in section IV-A, until it is satisfiable or until a predefined bound for the number of scan operations is reached. If the SAT instance is satisfiable, a sequence of CSU operations exists that satisfies the user specified constraints. If it is not satisfiable, the user constraints cannot be satisfied within the predefined scan access length.

In case the user constraints are satisfiable, the *satisfying assignment* (solution to the SAT instance) provides the state of all scan segments and control signals over the consecutive time frames. To find the scan-in data for the primary scan input of a network in the i-th CSU operation, the network's graph is traversed along the active scan path of frame $i-1$. The content of visited scan segments is acquired from the i-th time frame of the satisfying assignment. The concatenation of scan segment contents in the order of graph traversal forms the scan-in data for the i-th CSU operation. The solution is minimal with respect to the scan access length, i.e. the number of required CSU operations.

VI. Evaluation

The proposed method is evaluated on several benchmarks in two use cases: verification of scan architectures and pattern generation. As a SAT solver, MiniSat is used [21]. The experiments are run on an Intel Core2 CPU operating at 2.83 GHz.

A. Benchmark Circuits

To the best of our knowledge, there exist no openly available benchmark circuits with reconfigurable scan architectures. For the purpose of this study, we synthesize such scan architectures for the ITC'02 benchmarks, which are in widespread use for evaluation of test scheduling methods [22].

The ITC'02 benchmarks represent hierarchical systems with multiple modules (cores). The modules have a defined number of inputs, outputs, internal scan chains, and submodules (constituent cores). Table I presents their characteristics.

Design	Modules	Levels	Scan segments	Register bits
u226	10	2	40	1,416
d281	9	2	50	3,813
d695	11	2	157	8,229
h953	9	2	46	5,586
g1023	15	2	65	5,306
f2126	5	2	36	15,789
q12710	5	2	21	26,158
p22810	29	3	254	29,828
p34392	20	3	103	23,119
p93791	33	3	588	97,984
t512505	31	2	128	76,846
a586710	8	3	32	41,635

Table I
CHARACTERISTIC OF THE ITC'02 BENCHMARKS

We propose two reconfigurable scan architectures for the ITC'02 benchmarks: a design based on hierarchical gateways, and a custom hierarchical architecture. ICL models (as proposed by P1687) of the two scan architectures are generated automatically from the benchmark descriptions.

The first scan architecture is based on gateways called segment insertion bits (SIBs) in P1687 nomenclature. A SIB consists of a 1-bit configuration register and a scan multiplexer that either bypasses or connects the lower-level scan segment (or a scan network) to the higher-level scan chain, depending on the content of the configuration register. In the SIB-based architecture, the scan chain encompassing a single module is composed of several hierarchical gateways, as proposed in [19]. The SIBs provide configurable access

to the scan chains of the core, its submodules, as well as its inputs and outputs. Fig. 5 shows such a scan architecture for the top-level part of the p34392 benchmark.

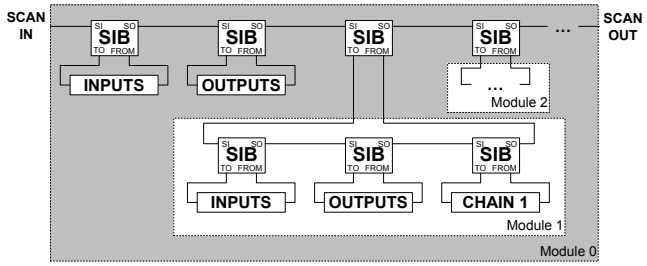

Figure 5. SIB-based scan architecture for the p34392 benchmark

The characteristic of the examined SIB-based architectures are listed in the first three columns of table II: The second column shows the number of required hierarchical gateways, whereas the third column gives the total number of required scan segments for SIBs, inputs, outputs, and internal scan chains.

In the custom hierarchical architecture, we distinguish two access modes: configuration access and data access. Configuration access mode allows to reconfigure the scan chain of a core by attaching or detaching its scan segments for inputs, outputs and internal scan chains, as well as subnetworks of constituent submodules. Fig. 6 shows the custom hierarchical architecture for the top-level part of the p34392 benchmark. The scan chain of each module starts with a 1-bit configuration register AM that distinguishes between configuration ($AM = 0$) mode, in which only the configuration registers (C) can be accessed, and data access mode ($AM = 1$). Once configured, this architecture is faster compared to the SIB-based scheme, as less control registers are present on the active scan path in the data access mode. However, in case of reconfiguration, the custom hierarchical architecture may require more CSU operations.

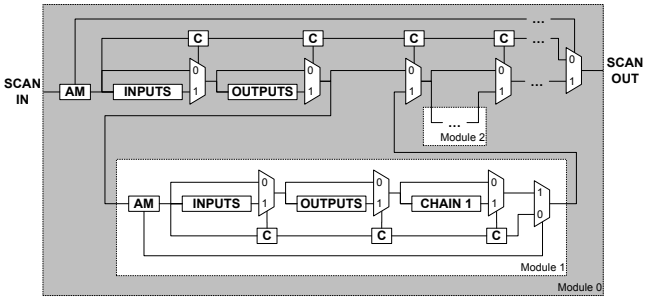

Figure 6. Custom hierarchical scan architecture for the p34392 benchmark

The properties of the examined custom hierarchical architectures are listed in the first three columns of table III. The second column shows the number of required scan multiplexers. In the third column, the total number of required scan segments is given, including the configuration registers.

B. Validation of Results

The results presented in the following sections are validated by cycle-accurate simulation in a commercial logic simulator. For this purpose, the benchmark ICL models are automatically translated to hardware Verilog models. The solution of network verification and pattern generation is used as stimuli for the primary scan input of a network. During simulation, assertions verify that the access to the scan network is valid, and that the user constraints are met.

C. Scan Network Verification

The integrity of the benchmark scan architectures is verified using the method of section V-A. It is formally proven that the designs are valid, i.e. all scan segments are observable and controllable in the default scan configuration, and that the default scan configuration can be restored within a bound of 100 time frames.

In column "Verification", table II and III present the results of network verification for the SIB-based and the custom architecture, respectively. Column "Access length" gives the average and maximal number of CSU operations (time frames) to access a single scan segment and restore the default scan configuration. Under "Clauses" we give the maximum number of clauses contained in the SAT instance after unrolling. Column t_{solve}^{max} is the maximum solve time for a single scan segment (iteration), whereas t_{total} is the total design verification time.

Although the size of unrolled SAT instances grows up to about 220,000 clauses, the maximum solve time for a single iteration is just 250 ms in the worst case, and 40 ms on average. This is due to the fact that the majority of clauses describes signal propagation with just two literals, which is efficiently handled by state-of-the-art SAT solvers. The total validation time is 16.5 s on average, and took less than 3 minutes for the largest design.

The verification of SIB-based architectures does not require the solver to backtrack since a solution is found by direct implications. Contrary to the SIB-based design, the custom hierarchical architecture may cause temporal conflicts and backtracking if the solver takes a wrong decision on the temporal access order to configuration registers. The average and maximum number of times the solver needs to backtrack is given in table III under "Conflicts".

D. Scan Pattern Generation / Pattern Retargeting

The proposed method is evaluated for pattern generation, as explained in section V-B. For each design, a series of 1000 random experiments is conducted. In a single experiment, 10 randomly chosen scan segments are accessed. The bound of unrolling is set to 100.

Design	SIB	Total scan segm.	Verification				Pattern generation			
			Access len. avg / max	Clauses max	t^{max}_{solve} [s]	t_{total} [s]	Access len. avg / max	Clauses max	t^{max}_{solve} [s]	t_{total} [s]
u226	50	90	2.3 / 3	8,872	0.01	0.5	2.6 / 3	8,896	0.02	7.5
d281	59	109	2.4 / 3	10,662	0.01	0.8	2.7 / 3	10,686	0.02	9.1
d695	168	325	2.5 / 3	31,466	0.03	6.8	2.7 / 3	31,490	0.03	27
h953	55	101	2.3 / 3	9,894	0.02	0.6	2.7 / 3	9,918	0.02	8.3
g1023	80	145	2.3 / 3	14,322	0.02	1.3	2.7 / 3	14,346	0.02	12
f2126	41	77	2.4 / 3	7,454	0.01	0.4	2.5 / 3	7,478	0.02	6.2
q12710	25	47	2.4 / 3	4,574	0.01	0.14	2.5 / 3	4,598	0.01	3.9
p22810	283	537	2.5 / 4	65,806	0.07	23	2.8 / 4	65,838	0.06	48
p34392	123	226	2.7 / 4	27,948	0.03	3.6	3.1 / 4	27,980	0.03	20
p93791	621	1,209	2.5 / 4	146,952	0.12	114	2.9 / 4	146,984	0.13	116
t512505	160	288	2.3 / 3	28,628	0.03	5.4	2.7 / 3	28,652	0.03	24
a586710	40	72	2.5 / 4	8,881	0.01	0.4	2.8 / 4	8,913	0.02	6.2

Table II
RESULTS FOR THE SIB-BASED SCAN ARCHITECTURE

Design	MUX	Total scan segm.	Verification					Pattern generation				
			Access len. avg / max	Clauses max	Conflicts avg / max	t^{max}_{solve} [s]	t_{total} [s]	Access len. avg / max	Clauses max	Conflicts avg / max	t^{max}_{solve} [s]	t_{total} [s]
u226	59	99	3.5 / 5	14,062	1.0 / 15	0.02	0.8	5.6 / 7	18,872	6.5 / 26	0.03	14
d281	67	117	3.7 / 5	16,366	1.7 / 9	0.02	1.2	5.7 / 7	21,956	6.4 / 21	0.03	16
d695	178	335	3.9 / 5	45,540	0.9 / 3	0.04	10.0	6.0 / 7	61,000	10.2 / 67	0.06	52
h953	63	109	3.6 / 5	15,302	0.9 / 7	0.02	1.0	5.7 / 7	20,532	7.4 / 26	0.02	15
g1023	94	159	3.6 / 5	22,492	0.9 / 9	0.02	2.2	5.9 / 7	30,152	8.7 / 25	0.03	24
f2126	45	81	3.7 / 5	11,218	1.1 / 11	0.01	0.6	5.6 / 7	15,068	4.9 / 18	0.02	11
q12710	30	51	3.6 / 5	7,228	0.9 / 5	0.01	0.3	5.7 / 7	9,728	5.1 / 16	0.02	7.7
p22810	311	565	3.9 / 7	104,044	1.3 / 24	0.09	32	6.0 / 10	143,565	11.5 / 58	0.14	92
p34392	142	245	4.4 / 7	46,004	2.6 / 24	0.04	6.1	6.9 / 10	63,520	14.2 / 65	0.06	43
p93791	653	1,241	4.1 / 7	224,852	1.7 / 38	0.24	176	6.0 / 9	281,758	17.5 / 97	0.37	229
t512505	191	319	3.6 / 5	45,302	0.9 / 7	0.04	8.9	5.7 / 7	60,672	11.9 / 50	0.06	48
a586710	47	79	4.0 / 7	15,016	1.9 / 15	0.01	0.6	6.3 / 10	20,787	8.2 / 61	0.02	13

Table III
RESULTS FOR THE CUSTOM HIERARCHICAL SCAN ARCHITECTURE

In column "Pattern generation", table II and III present the results of pattern generation in a similar way as was done for verification. In contrast to verification, here 10 randomly chosen scan segments are accessed in a single experiment. Column t_{total} shows the pattern generation time for the total of 1000 experiments.

In the worst case, a scan access with 10 targets requires 370 ms for solving for p93791. On average over all designs, 35 s are required to generate scan-in data for the total of 1000 scan accesses. Unlike SIB-based designs, the custom hierarchical architectures cause conflicts that require the solver to backtrack as discussed above.

E. Performance Analysis

The problem of pattern generation for reconfigurable scan networks can be reformulated to facilitate the solution with general purpose formal reasoning tools. In the following, we redefine the pattern generation problem as a model checking problem, and provide experimental results obtained with a commercial model checker.

The model checker is used to prove that the user specified access to scan segments (user constraints) is not satisfiable, or to generate a valid access sequence as a counter example. The system that is subject to model checking is the hardware model of the scan network. Additional assumptions are imposed on the model to enforce valid CSU operations. The specification is an assertion specifying that the user constraints are *never* met, i.e. there exists no sequence of CSU operations that satisfies the user specified read or write operation. For instance, for a read operation on scan segment S1, the specification states that S1 is never on the active scan path, which is in CTL: $\neg EF(sel(S1))$, or equivalently $AG(\neg sel(S1))$.

If the model meets the specification, no valid scan access exists that satisfies the user constraints. Otherwise, if the specification is not met, the user constraints can be satisfied. In this case, the by-product of model checking in form of a *counter example* provides the required scan-in sequence.

We perform several experiments using a state-of-the-art commercial model checker for digital circuits. The tool accepts a Verilog model of a design, while assumptions and assertions are specified in PSL.

The experiments are based on the largest benchmark (p93791) with custom scan architecture. In each experiment, 10 user constraints are specified for random read/write access to scan segments. A solving time limit is set to 1

hour. Table IV shows the time that is required to solve the pattern generation problem with the model checker (second column) and the proposed approach (third column). The model checker exceeds the time limit in two experiments. In the remaining experiments its solving time varies widely from 12 up to 364 seconds. In contrast, the proposed modeling approach is successful in all the experiments and exhibits much more stable run-times below 0.21 s. This result clearly shows that the proposed domain-specific modeling provides a great performance improvement over a mapping to a general model checking problem.

Exp. No.	Model Checking	Proposed	Speedup
1	79 s	0.19 s	415x
2	364 s	0.13 s	2800x
3	12 s	0.12 s	100x
4	38 s	0.14 s	271x
5	45 s	0.12 s	375x
6	27 s	0.21 s	129x
7	28 s	0.20 s	140x
8	-	0.21 s	-
9	-	0.14 s	-
10	54 s	0.13 s	415x

Table IV
PERFORMANCE COMPARISON IN RANDOM EXPERIMENTS

VII. CONCLUSION

Reconfigurable scan networks allow flexible and scalable access to on-chip infrastructure. The design complexity due to hierarchies, IP reuse and complex constraints requires novel EDA tools for optimization, verification and pattern generation. The proposed modeling allows the thorough analysis of these scan networks including the combinational and sequential constraints by mapping the problem to formal SAT reasoning. The method is robust compared to simple heuristic approaches, as it exhaustively explores the search space. In future, it may serve as the basis to debug design errors by conflict analysis or for advanced property checking. The experiments demonstrate the applicability to larger benchmarks and the performance improvement over a mapping to a model checking problem.

ACKNOWLEDGEMENTS

Parts of this work were supported by the German Research Foundation (DFG) under grants WU 245/11-1 and WU 245/13-1.

The authors thank Christian Zoellin for many insightful discussions about the IEEE Std. P1687.

REFERENCES

[1] S. Narayanan and M. A. Breuer, "Reconfigurable Scan Chains: A Novel Approach To Reduce Test Application Time," in *Proc. Int'l Conf. on Computer-Aided Design (ICCAD)*, 1993, pp. 710–715.

[2] S. Samaranayake, E. Gizdarski *et al.*, "A Reconfigurable Shared Scan-in Architecture," in *Proc. VLSI Test Symp. (VTS)*, 2003, pp. 9–14.

[3] B. Arslan and A. Orailoglu, "Test Cost Reduction Through A Reconfigurable Scan Architecture," in *Proc. Int'l Test Conf. (ITC)*, 2004, pp. 945–952.

[4] M. E. Imhof, C. G. Zoellin *et al.*, "Scan Test Planning for Power Reduction," in *Proc. Design Automation Conf. (DAC)*, 2007, pp. 521–526.

[5] D. Xiang, Y. Zhao *et al.*, "A Reconfigurable Scan Architecture With Weighted Scan-Enable Signals for Deterministic BIST," *IEEE Trans. CAD*, vol. 27, no. 6, pp. 999–1012, 2008.

[6] B. B. Bhattacharya, S. C. Seth, and S. Zhang, "Double-Tree Scan: A Novel Low-Power Scan-Path Architecture," in *Proc. Int'l Test Conf. (ITC)*, 2003, pp. 470–479.

[7] B. Nadeau-Dostie, S. Adham, and R. Abbott, "Improved Core Isolation and Access for Hierarchical Embedded Test," *IEEE Design & Test of Computers*, vol. 26, no. 1, pp. 18–25, 2009.

[8] E. J. Marinissen, R. G. J. Arendsen *et al.*, "A Structured and Scalable Mechanism for Test Access to Embedded Reusable Cores," in *Proc. Int'l Test Conf. (ITC)*, 1998, pp. 284–293.

[9] S. Koranne, "Design of Reconfigurable Access Wrappers for Embedded Core Based SoC Test," *IEEE Trans. VLSI Syst.*, vol. 11, no. 5, pp. 955–960, 2003.

[10] B. Eklow and B. Bennetts, "New Techniques for Accessing Embedded Instrumentation: IEEE P1687 (IJTAG)," in *Proc. European Test Symp. (ETS)*, 2006, pp. 253–254.

[11] M. Abramovici, P. Bradley *et al.*, "A Reconfigurable Design-for-Debug Infrastructure for SoCs," in *Proc. Design Automation Conf. (DAC)*, 2006, pp. 7–12.

[12] N. Stollon, "IEEE P1687 - IJTAG," in *On-Chip Instrumentation*. Springer US, 2011, pp. 137–144.

[13] J. Remmers, M. Villalba, and R. Fisette, "Hierarchical DFT Methodology - A Case Study," in *Proc. Int'l Test Conf. (ITC)*, 2004, pp. 847 – 856.

[14] A. Benso, S. Di Carlo *et al.*, "IEEE Standard 1500 Compliance Verification for Embedded Cores," *IEEE Trans. VLSI Syst.*, vol. 16, no. 4, pp. 397–407, 2008.

[15] P. Horstmann and E. Stabler, "Computer Aided Design (CAD) Using Logic Programming," in *Proc. Design Automation Conf. (DAC)*, 1984, pp. 144–151.

[16] 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.

[17] E. Eichelberger and T. Williams, "A Logic Design Structure for LSI Testability," in *Proc. Design Automation Conf. (DAC)*, 1977, pp. 462–468.

[18] F. Zadegan, U. Ingelsson *et al.*, "Test Time Analysis for IEEE P1687," in *Proc. Asian Test Symp. (ATS)*, 2010, pp. 455–460.

[19] F. Zadegan, U. Ingelsson *et al.*, "Design Automation for IEEE P1687," in *Proc. Design, Automation Test in Europe Conf. (DATE)*, 2011, pp. 1412–1417.

[20] S. Eggersglüß, G. Fey *et al.*, "Combining Multi-Valued Logics in SAT-based ATPG for Path Delay Faults," in *Proc. Int'l Conf. on Formal Methods and Models for Codesign*, 2007, pp. 181–187.

[21] N. Eén and N. Sörensson, "An Extensible SAT-solver," in *Proc. Theory and Applications of Satisfiability Testing (LNCS)*, vol. 2919, 2003, pp. 502–518.

[22] E. Marinissen, V. Iyengar, and K. Chakrabarty, "A Set of Benchmarks for Modular Testing of SOCs," in *Proc. Int'l Test Conf. (ITC)*, 2002, pp. 519–528.