Access Time Minimization in IEEE 1687 Networks

Krenz-Baath, René; Ghani Zadegan, Farrokh; Larsson, Erik

Published in:
[Host publication title missing]

DOI:
10.1109/TEST.2015.7342408

2015

Access Time Minimization in IEEE 1687 Networks

René Krenz-Baath*, Farrokh Ghani Zadegan†, Erik Larsson†
*Hamm-Lippstadt University of Applied Sciences, Hamm, Germany
Rene.Krenz-Baath@hshl.de
†Lund University, Lund, Sweden
ghanizadegan@ieee.org, erik.larsson@eit.lth.se

***Abstract*—IEEE 1687 enables flexible access to the embedded (on-chip) instruments that are needed for post-silicon validation, debugging, wafer sort, package test, burn-in, printed circuit board bring-up, printed circuit board assembly manufacturing test, power-on self-test, and in-field test. At any of these scenarios, the instruments are accessed differently, and at a given scenario the instruments are accessed differently over time. It means the IEEE 1687 network needs to be frequently reconfigured from accessing one set of instruments to accessing a different set of instruments. Due to the need of frequent reconfiguration of the IEEE 1687 network it is important to (1) minimize the run-time for the algorithm finding the new reconfiguration, and (2) generate scan vectors with minimized access time. In this paper we model the reconfiguration problem using Boolean Satisfiability Problem (SAT). Compared to previous works we show significant reduction in run-time and we ensure minimal access time for the generated scan vectors.**

Keywords*—IEEE Std. 1687, retargeting, upper bound calculation, access time minimization

I. Introduction

IEEE 1687 (IJTAG) [1] enables flexible access, mainly through the JTAG test access port (TAP) [2], to the on-chip instruments that are needed for post-silicon validation, debugging, wafer sort, package test, burn-in, printed circuit board bring-up, printed circuit board assembly manufacturing test, power-on self-test, and in-field test. At each of these scenarios, IEEE 1687 makes it possible to include only the desirable instruments in the scan-path. Hence, there is a need to reconfigure the IEEE 1687 network per scenario. There is also a need to frequently reconfigure the scan-path within a given scenario, as the set of instruments that are to be included on the scan-path changes over time. For each change in accessing the instruments, the IEEE 1687 network has to be reconfigured from accessing one set of instruments to accessing another set of instruments. In this paper this is called a retargeting step. As there are many reconfigurations, it is important to optimize the retargeting step. The efficiency of a retargeting step is given by the CPU time needed to find the required scan vectors, and the effectiveness is given by the access time of the generated scan vectors.

Previous work has addressed the problem with the retargeting step by proposing Boolean satisfiability problem (SAT) modeling for IEEE 1687 networks [3] and effective retargeting heuristics [4]. A key to optimize the retargeting step is to (1) minimize the number of SAT calls, and (2) find bounds limiting the search space without excluding the optimal solution.

In this paper, we propose a SAT modeling and an improved bound calculation. We have implemented our method, as well as a method in a previous work. The result is significant improvement on efficiency (run-time of the algorithm to find the new configuration) while guaranteeing minimal access time of the generated scan vectors, compared to previous work.

The rest of this work is organized as follows. We start by an introduction to IEEE 1687 and basics of retargeting, as well as to SAT and its current applications in the field (Section II). Related work will be discussed in Section III where we also point out in what ways we have improved the previous work. Section IV will present our upper bound calculation method. The calculated upper bound is used in Section V, where we describe our modeling approach, in order to calculate the optimal access time for the considered networks.

II. Background

In this section, the relevant hardware features of IEEE 1687 will be introduced (Section II-A), and the retargeting concept will be explained (Section II-B). Moreover, as we have modeled the IEEE 1687 retargeting as Boolean satisfiability problem (SAT), Section II-C will give a brief introduction into SAT.

A. Instrument Access Infrastructure (Network)

A strong feature in IEEE 1687 networks is the possibility of dynamic reconfiguration, which allows for reduction of instrument access time by varying the length of the scan-path to include only those instruments in the path which are needed for current session. To enable variable-length scan-paths in IEEE 1687 networks, a *ScanMux* control bit is used, which is a shift-update register that can be placed anywhere on the scan-path to configure one or more scan multiplexers (ScanMux components). Fig. 1(a) shows ScanMux control bits C_1 and C_2 used to configure a network of two instruments. To program the control bits to any desired configuration, the right values should be placed in their shift cells (denoted by **S**) during the *Shift* phase, and copied to their parallel latch (denoted by **U**) during the *Update* phase. We will use the symbol in Fig. 1(b) to represent a ScanMux control bit in the rest of this paper.

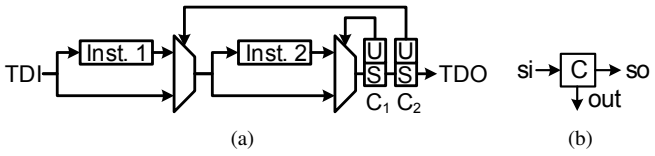

Fig. 1. ScanMux control bit: (a) a network of two instruments, configured by ScanMux control bits C_1 and C_2, (b) corresponding symbol

To access the network of instruments from the chip boundary, IEEE 1687 specifies the JTAG TAP as the primary interface. Interfacing is performed by connecting the first level of the IEEE 1687 network as a custom TDR to the JTAG circuitry. Since the JTAG TAP FSM is primarily used to operate IEEE 1687 networks, performing each cycle of network configuration involves going through the capture, shift, and update states in the FSM, which is referred to as a CSU cycle [1]. As an example, Fig. 2 illustrates a small IEEE 1687 network consisting of three instruments (namely a DFT instrument in the first hierachical level, and a sensor and a debugging feature in the second hierachical level) and six ScanMux control bits (C_1–C_6). The instruments are interfaced to the scan-path through shift-registers with parallel I/O. To access the instruments, ScanMux control bits should be programmed to include the required shift-registers into the scan-path. For example, to access only the DFT feature, C_1 and C_2 should be set to logic value "1", and C_3 should be set to "0" to bypass (via input 0 of mux M_3) the network segment containing the Sensor and Debug instruments, as well as C_4, C_5, and C_6 components.

It can be immediately noticed from the network in Fig. 2 that reconfiguring the network to the desired configuration might need several CSU cycles (CSUs). For example, assuming an initial configuration of $C_1 = \ldots = C_6 = 0$, accessing the Debug instrument needs two cycles of shift and update. In the first cycle, only C_1, C_2, and C_3 are accessible and by setting $C_2 = 0$ and $C_1 = C_3 = 1$, C_4, C_5, and C_6 become accessible. It is in the second cycle when C_4, C_5, and C_6 can be configured to the right values, i.e., $C_5 = 0$ and $C_4 = C_6 = 1$, so that the Debug instrument becomes accessible.

B. Description Languages and Retargeting

IEEE 1687 introduces two description languages, namely Instrument Connectivity Language (ICL) and Procedural Description Language (PDL). ICL is used to describe the network, that is, how the instruments are connected to the JTAG TAP. PDL is used to describe the operation of instruments at their terminals. PDL commands allow to perform read/write operations on the instrument shift-registers and configurable components, as well as to wait for an instrument (such as a BIST engine) to finish its operation.

Given the PDL of each instrument, a retargeting tool generates scan vectors to configure the network and transport the required data bits from the JTAG TAP to/from the instruments' shift-registers. A retargeting tool relieves the designer from dealing with network configuration (i.e., writing the PDL to configure ScanMux Control bits directly). For example, assuming that the goal is to read the value from the sensor instrument in Fig. 2, the PDL developer might simply use a write command to activate the sensor, a wait command to wait for the sensor to capture the value, and a read command to read the captured value out. It is then the task of the retargeting tool to generate one scan vector to configure C_1, C_2, and C_3, one vector to configure C_4, C_5, and C_6, one vector to write to the enable bit in the sensor's shift-register, a wait cycle of enough length, and finally one vector to scan the captured value out.

In its basic form, a PDL script is a sequence of iApply groups. In each iApply group, there are a number of read and write operations to the registers in the network (*setup* commands). These read/write operations take effect upon encountering an iApply command (an *action* command). A retargeting step will then be to generate a number of scan vectors to (1) change the configuration of the network (from its current state) to a configuration in which the specified registers are accessible, and (2) to perform the read/write operation. These vectors are then applied to the network through a number of CSUs. A complete retargeting flow will then be a number of such retargeting steps.

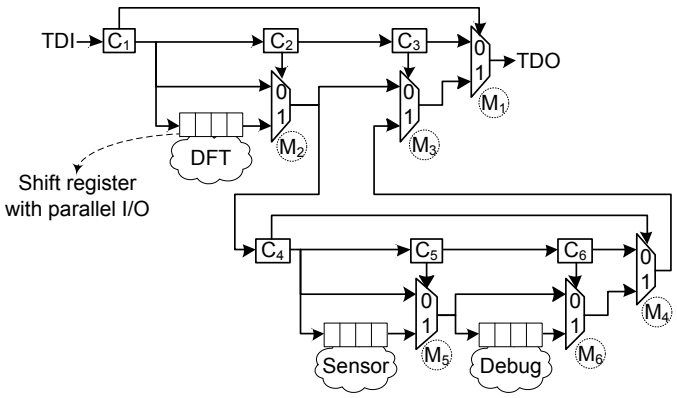

Fig. 2. An IEEE 1687 network with three instruments inside a chip

For complex IEEE 1687 networks and especially for long PDL scripts, it becomes desirable to both speed up the retargeting process and to generate effective scan vectors which are optimal with regards to the application time. To achieve this goal, a first step would then be to optimize the basic retargeting step for both run-time efficiency and effectiveness of the generated vectors. There have been a number of works addressing retargeting for an IEEE 1687 network [3]–[8]. So far only [3]–[5] have addressed the issues of efficiency and effectiveness in retargeting. What distinguishes [5] from the other works is addressing the efficiency of retargeting when applying interactive PDL (PDL Level-1 which supports programming language constructs such as conditions and loops) with the help of hardware acceleration. Therefore, the only works that have so far addressed efficiency and effectiveness for a basic retargeting step are [3], [4].

C. Boolean Satisfiability Problem (SAT)

The retargeting approach proposed in this paper employs a SAT-based reasoning engine [9]. The Boolean satisfiability problem is the problem of finding an assignment satisfying some given Boolean formula. Typically the Boolean formula is expressed in the *conjunctive normal form* (CNF), denoted Φ, which is the conjunction of a set of clauses C, where each clause $c \in C$ is a disjunction of Boolean literals, $c = l_1^c \vee \ldots \vee l_n^c$.

The satisfiability problem is among the first problems proven to be NP-hard. Recent developments in the area of SAT, such as conflict-based learning, conflict analysis [9]–[11] as well as powerful preprocessing techniques [12], led to the integration of SAT-solvers in almost all areas of the electronic design automation (EDA) industry [13]–[15].

Further extensions of SAT-based reasoning engines enable its application to multi-valued problems [16] or 0-1-ILP problems [17], [18]. These extension enabled further applications for example in the area of test set optimization [19] or for an optimized collapsing of cell-aware fault sets [20].

III. Prior Work

Verification and scan vector generation (retargeting) for reconfigurable scan networks were presented in [3]. The work in [3] models general reconfigurable scan networks using a structural SAT model which captures any arbitrary configuration of the network. In a typical retargeting step, several configuration cycles should be performed to take the network from an initial configuration to a final (requested) configuration—in which the shift-registers of the required instruments become part of the active TDI to TDO scan-path. Therefore, to capture all the configuration cycles, the SAT model is unrolled for a number of time frames. Each of the time frames corresponds to an atomic CSU sequence. That is, e.g., each individual clock cycle spent on shifting input data is not considered a separate configuration step, rather the whole cycle of capturing, shifting, and updating is seen as one. The state of each scan segment (i.e., a single-bit or multi-bit shift-register) in each configuration time frame is then used as input data that should be shifted in and applied (by going through the update phase) for the transition from a frame to the next one. The sequence of such input vectors is actually what a retargeting tool should find out in order to take the circuit from an initial configuration to a requested configuration.

The use of the above-mentioned scheme requires the algorithm to receive as input the number of times it should unroll the model, i.e., the number of allowed CSUs. The choice of the number of CSUs has a crucial impact on the obtained solution. That is, if the allowed number of CSUs is too small, none of the possible solutions might be reachable from the current configuration. Moreover, given that some solutions might be better than the others w.r.t. access time (in terms of test clock cycles), a too small number of CSUs might exclude those better solutions from the solution space. Therefore, a key to effective retargeting (i.e., generating scan vectors which are optimal w.r.t. access time in terms of clock cycles) is the right upper bound on the number of CSUs. On the other hand, if the number of allowed CSUs is too large, the generated model becomes unnecessarily large resulting in decreased run-time efficiency.

In Section III-A it is explained how in prior work the choice of the upper bound on the number of CSU cycles is addressed. In Section III-B, the modeling of the network into SAT formulas as done in prior work will be elaborated on.

A. Upper Bound Calculation

The work in [3] does not present an upper bound derivation method for the number of required CSU cycles and assumes that the user specifies a maximum allowable number of frames. Moreover, the generated scan vectors are not optimal regarding instrument access time (in clock cycles). To address these issues, [4] presents an upper bound for the number of configuration time frames. The calculation of upper bound on the number of frames, as is presented in [4] can be explained as follows. The total access time can be formulated as $t = 2n + \sum_{i=1}^{n} L_i$, where n is the number of frames, 2 represents the number of clocks spent on applying the stimuli and capturing the responses for each frame, and L_i represents the length of the scan-path for frame i. The upper bound for n, denoted by n_{bound}, is presented as

$$n_{\text{bound}} < \lceil \texttt{Cycles}_n/2 \rceil, \tag{1}$$

where $\texttt{Cycles}_n$ is the minimum access time achievable with n frames. According to the work in [4], finding the global minimum is then an iterative process in which the bound is iteratively lowered as we find solutions with smaller access times (i.e., smaller than $\texttt{Cycles}_n$ which was originally found).

Given that in real-life circuits, the access time might be in the order of thousands of clock cycles, the calculated bound (1) will not be helpful in practice. The reason is that, as discussed in [4], finding the optimal solution involves heavy computations required to search the solution space, which is limited by the upper bound on the number of frames. If this upper bound is very high (that is, hundreds or even thousands of frames), the time that it takes to find the optimal solution will be very long. Therefore, the authors in [4] propose a heuristic for effective (with regards to access time) retargeting. The proposed algorithm initially searches for the minimum number of required frames, and from that point continues the search by allowing a limited number of extra frames until it either reaches a local minimum or exhausts the allowed extra frames.

In this work (Section IV), we improve the work in [4] by proposing an upper bound calculation method which helps to achieve optimal solution for the MUX-based IEEE 1687 networks described in [4].

B. Reasoning in IEEE 1687 Networks

This subsection provides an overview of previous work in verification and retargeting approaches in reconfigurable scan networks. In particular we focus on aspects regarding IEEE 1687 networks.

The initial paper introducing a formal representation to verify controllability and observability of components within some reconfigurable scan network as well as retargeting predefined test patterns has been published in [3]. The authors describe an approach to model a reconfigurable scan network in such a way that a SAT-based reasoning engine [21] can be applied to perform property checking as well as retargeting of test patterns. First combinational dependencies with respect to the controllability and observability of instruments and other scan segments are modeled in CNF for a single time frame. Next, the above representation is extended to model several time frames and hence to enable the retargeting of PDL commands in IEEE 1687 networks (which requires a number of sequential scan vectors to be generated). This approach is similar to state-of-the-art bounded model checking techniques applied in formal verification [22]. Later, the authors proposed an extension of the previously discussed approach to enable the merge of several patterns as well as to reduce of the overall length of the resulting scan vector [4]. In the following, we will discuss the network modeling in [4] in details.

A scan network is transformed into a directed graph $G = (V, E)$ consisting of a set of vertices V and a set of connecting edges $E \subseteq V \times V$ representing individual network components and network connections, respectively. Furthermore, the set of vertices $V_P \subseteq V$ represents primary scan-in and scan-out ports. The vertex sets $V_S \subseteq V$ and $V_A \subseteq V$ represent the set of scan elements and the set of multiplexers and fanouts, respectively.

The goal of a retargeting process is to read or write to some instrument within the network. In order to access that instrument the scan network has to be configured in a way that the desired instrument is part of an active scan-path. Assume some scan element represented by a vertex $v \in V_S$ being part of an active scan-path $A \subseteq V$, where $V_P \cap A \neq \emptyset$ and for each $v \in A \cap (V_S \cup V_A)$ there exist some successor $s \in V \setminus v$ and some predecessor $p \in V \setminus v$ and there exists an edge $e \in E$ connecting v and s as well as another edge $d \in E$ connecting v and p. The proposed model requires that at least one successor s and at least one predecessor p also being part of the active path, $\{s, p\} \subset A$. For every vertex v to be member of an active path A, a corresponding predicate function $sig(v)$ has to evaluate to $true$. Following the requirements discussed above, $sig(v)$ being true requires that the predicate functions of p and s also need to be true, $sig(v) \rightarrow sig(p) \wedge sig(s)$. The predicate function of every vertex v within an active scan-path A has to evaluate to true. In order to generate a complete model of the scan network representing every possible scan-path within the network, for vertices having more than a single predecessor or several successors the above rule is extended such that $sig(v) \rightarrow \exists_{p \in P(v)}\ sig(p) \wedge \exists_{s \in S(v)}\ sig(s)$, where $P(v)$ and $S(v)$ denote the set of predecessors and successors of v in V, respectively.

If some successor s of some vertex v models a multiplexer in the IEEE 1687 network, then the predicate function of v is extended in order to enforce the corresponding select signal to activate the required MUX-branch. The additional constraint is formulated as follows:

$$sig(s) \rightarrow sel(s, v), \tag{2}$$

where $sel(s, v)$ denotes the corresponding select signal and its value required to activate the MUX-branch coming from vertex v.

If some vertex v corresponds to a multiplexer in the scan network, then the predicate function of v is extended such that for every possible predecessor $p \in P(v)$ a constraint is added to $sig(v)$ which is formulated as follows:

$$sig(p) \rightarrow sel(v, p), \tag{3}$$

where $sel(v, p)$ denotes the select signal of vertex v and its value to activate the path between v and p.

The set of constraints described above models the combinational dependencies within an IEEE 1687 network, which need to be satisfied in order to generate an active scan-path between some TDI and some TDO containing a set of active scan segments.

Considering the network example shown in Figure 3(a), it is obvious that depending on the Boolean values programmed into the ScanMux control bits C_1 and C_2 there might be several CSUs required to generate an active scan-path containing segment S_1. Hence the problem of retargeting PDL commands is a sequential reasoning problem, which can be solved by applying SAT-based bounded model checking initially described in [22]. The basic concept of SAT-based bounded model checking is to generate a number of copies of the combinational components of a circuit, where for every signal an individual variable per time frame is introduced. The values stored in the memory elements of one time frame are copied to the corresponding memory element of the next time frame. The modification of the stored value is enabled if the update signal of the corresponding memory element is active in the current time frame. With respect to the discussed retargeting problem it is required that a ScanMux control bit is part of an active scan-path in order to be programmed.

Assume that the task is to activate segment S_1. Let us further assume that ScanMux control bits C_1 and C_2 are set to '1' and '0', respectively. This implies that the currently active scan-path starts at TDI, passes the ScanMux control bit C_1, the multiplexer M_2, the multiplexer M_1, and finally reaches TDO. In order to activate segment S_1 it is necessary to first generate an active scan-path containing C_2 to modify the select value of multiplexer M_1 enabling the activation of S_1 in a subsequent CSU. Hence the first scan vector applied sets value in C_1 to '0'. The resulting active scan-path contains both ScanMux control bits C_1 and C_2. Due to that, the following scan vector can modify the ScanMux control bit values such that $C_1 = 1$ and $C_2 = 1$, enabling an active scan-path in the following time frame containing segment S_1.

The above example demonstrates that depending on the initial values in $C_1 = 1$ and $C_2 = 1$ finding a solution to activate S_1 is an unsatisfiable problem for a model containing less than two time frames. Hence for every retargeting problem there exists a minimum number of time frames or sequential depth that is required to successfully retarget some PDL command depending on a given initial state of the network. In [4] the authors search for the minimum number of time frames, CSU_{min}, by performing several SAT runs on the problem using an incremental number of time frames until a solution is found. The resulting number of minimum time frames corresponds to the minimum number of CSUs required to retarget a given set PDL commands.

A further aspect of reducing the overall access time is to minimize the number of shift cycles. The authors in [4] propose to transform the earlier formulated SAT problem described in CNF into a pseudo-Boolean problem in order to generate a solution requiring a minimal number of shift cycles. Pseudo-Boolean problems are also known as *0-1 integer linear programming* (0-1-ILP) problems. These 0-1-ILP problems are seen by the linear programming community as just a domain restriction on general linear programming. Within a pseudo-Boolean representation it is possible to formulate a cost function which is applied in order to find a solution for a given problem which is minimal with respect to this cost function. The proposed cost function for a retargeting problem over n time frames is formulated as follows:

$$Cycles = \sum_{i=0}^{n} \sum_{y=0}^{m} sig(y, i), \tag{4}$$

where n denotes the number of CSUs and m the number of scan elements. The function $sig(y, i)$ evaluates to 1 if scan element y is contained in an active scan-path during the ith

CSU. Otherwise $sig(y, i)$ evaluates to 0; By finding a valid solution of the retargeting problem, guided by the described cost function it is ensured to obtain a retargeting solution requiring the minimal number of test access time for the minimum number of CSUs.

Additionally the authors state in [4] that the minimum number of shift cycles might be obtained by a retargeting scenario performing more CSUs than the minimum number of CSUs described above. Therefore the authors run up to six further minimizations on problem descriptions containing an incremental number of time frames, meaning maximal $CSU_{min} + 6$ time frames. As described in Subsection III-A and according to [4] adding a constant number of time frames does not necessarily ensure finding the global minimum of a given retargeting problem.

IV. Upper Bound Computation

As discussed in Section III, retargeting is effective when the application of the generated scan vectors results in the least number of test clock cycles, which requires that the whole solution space be explored during retargeting. It was also mentioned that the number of allowed CSU cycles should be chosen such that neither any solution is removed from the solution space (which happens if not enough CSUs are allowed), nor is the run-time efficiency decreased (which happens if too many CSUs are allowed). Therefore, there is a need to find the upper bound on the number of CSU cycles that the search algorithm can use in order to reconfigure the network from any initial state to any target configuration, while considering all the possible solutions and choosing the one that results in the least access time in terms of test clock cycles. In this section, we present an upper bound calculation method for a subset of IEEE 1687 networks described in prior work [4], referred to as MUX-based networks. It should be mentioned that in [4], experiments are performed also on another architecture which is referred to as SIB-based architecture. However, the authors state that for SIB-based architecture, retargeting reduces to a simple decision problem, and therefore, in this work we only focus our attention on the more challenging MUX-based architecture.

Fig. 3(a) shows part of an IEEE 1687 network (referred to as MUX-based in [4]), in which M_2 (controlled by C_2) is used to bypass an IEEE 1687 network segment S_1, and M_1 (controlled by C_1) is used to select between C_2 and the mux M_2. The select (enable) and control (capture, shift, update) signals are not shown in the figure. The select signal is used to gate the control signals so that at any time a unique scan-path is activated. In the shown network, the assumption is that the select signal is connected such that only the scan-path connected to the selected input of a mux is activated. For example, if the aim is to activate S_1, both C_1 and C_2 should be set to logic value "1" so that the active scan-path goes from TDI to TDO via S_1, M_2 , and M_1. In the illustrated network, no matter what values C_1 and C_2 are initially set to, any desired configuration of these two components can be achieved within at most two CSU cycles. For example, assume that initially $C_1 = C_2 = 0$ and that the aim is to access S_1. In this case, both C_1 and C_2 are on the active scan-path (C_1 is always on the active scan-path and C_2 is on the active scan-path since $C_1 = 0$) and can be programmed to any desired

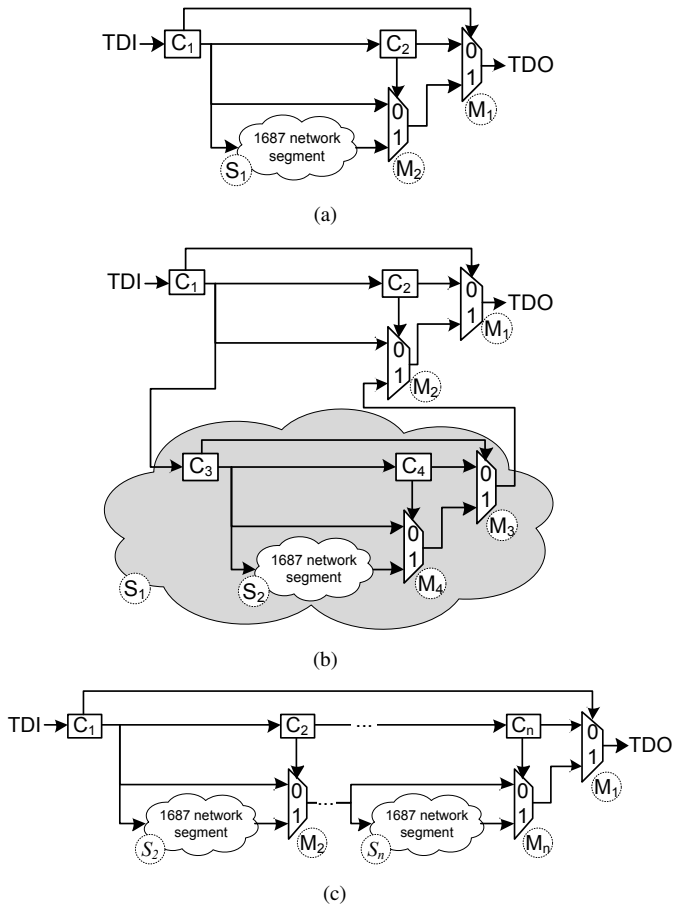

Fig. 3. Examples of MUX-based network used for computation of upper bound: (a) a network where the "1687 network segment" (i.e., S_1) can be an instrument shift-register or a network. (b) a network based on (a) where the "1687 network segment" is replaced with entire network in (a). (c) a generic structure in which "1687 network segment" parts can be replaced by the network in (a) to create a hierarchical MUX-based network.

configuration value with only one CSU cycle. If, however, initially $C_1 = 1$ and $C_2 = 0$, only C_1 is on the active scan-path (which goes from TDI to TDO via C_1, input 0 of M_2, and input 1 of M_1). Therefore, in order to achieve a configuration in which $C_1 = C_2 = 1$, first a CSU cycle is needed to set $C_1 = 0$ so that the active scan-path goes through C_2, and then a second CSU cycle is needed to set $C_1 = C_2 = 1$. It should be noted that the maximum two CSU cycles for configuration of C_1 and C_2 components is independent of the topology of S_1. That is, no matter if S_1 is an instrument shift-register or a network, it is possible to activate S_1 within maximum two CSU cycles.

If in Fig. 3(a), S_1 is replaced by a network similar to Fig. 3(a), the resulting network is as shown in Fig. 3(b). In this network, in order to activate S_2, all four ScanMux control bits C_1–C_4 should be set to "1". Similar to the argument in the previous paragraph, it can be argued that no matter what the initial configuration is, it takes at most four CSU cycles to activate S_2. This can be proven by considering that C_1 is always on the active scan-path. Therefore, if $C_2 = 0$ and is not on the active scan-path (i.e., $C_1 = 1$), it takes at most two CSU cycles to set $C_1 = C_2 = 1$ (by first setting $C_1 = 0$ to access C_2, and then setting $C_1 = C_2 = 1$). Once $C_1 = C_2 = 1$, S_1 is

activated and the same argument as the above paragraph can be used to say that it will take at most two extra CSU cycles to activate S_2—hence maximum four CSU cycles. If we again replace S_2 with another instance of the network in Fig. 3(a), we will need at most six CSU cycles to activate the "1687 network segment" in the resulting network. Therefore, if we consider each of these replacements of "1687 network segment" with the network in Fig. 3(a), as adding another hierarchical level, it can be concluded that for each additional level, two extra CSU cycles are needed.

Finally, Fig. 3(c) shows a generic MUX-based IEEE 1687 network in which C_2 to C_n are used to bypass their associated segments (i.e., S_2–S_n) and C_1 is used to select between the scan-path going through C_2–C_n and the path that goes through the muxes M_2–M_n. Since in this case, C_2–C_n receive their select signals from C_1 (meaning that they receive capture, shift, and update signals simultaneously), and since there is no functional correlation between C_2–C_n components, all of them can be configured independently but at the same time. Therefore, the argument for Fig. 3(a) applies to this network, too. That is, as C_1 is always on the active scan-path, it is possible to activate any of the segments S_2–S_n from any given initial configuration of the network, with at most two CSU cycles. By the same token, the segments S_2–S_n can also be independently and simultaneously configured. Now, from the argument in the above paragraph, we know that if any of S_2–S_n segments has multiple hierarchical levels, for each additional hierarchical level, two extra cycles are needed for configuration of the corresponding segment. Since we can access all the segments S_2–S_n at the same time, the upper bound on the total number of CSU cycles needed to explore all the possible configurations of the network, is the maximum hierarchical depth found in the generic network shown in Fig. 3(c), multiplied by two.

V. Improved Modeling and Optimal Retargeting

In this section we introduce the notion of *perfect networks* (PNs) within reconfigurable scan networks. Functional and structural properties of PNs are described and examples are provided. Next we discuss how PNs can simplify the modeling of reconfigurable scan networks and in particular IEEE 1687 networks. Finally it is described how the reductions obtained by using PNs together with the observations formulated in Section IV can be applied to ease the retargeting process in IEEE 1687 networks in order to enable a minimum with respect to the number of CSUs and the overall number of shift cycles. In contrast to the approach presented in [4] our method ensures optimality with respect to minimal access time for a subset of IEEE 1687 networks (described as MUX-based networks in [4]).

A. Perfect Networks in IEEE 1687 Networks

A typical IEEE 1687 network is shown in Fig. 4. The parts of the network labeled PN_1, PN_2, and PN_3 depict a reappearing network structure, which we refer to as a PN. The key properties of such a network structure are that there exists a single test data input, a single test data output and a set of control bits, such that every possible assignment of these control bits establishes an active path between the test data input and the test data output.

Applying the notation introduced in Subsection III-B it is possible to apply structural properties to formulate rules defining the data input and data output as well as control inputs of a PN.

Assuming a reconfigurable scan network as described in Subsection III-B then the data input vertex of a PN is dominated by the data output of this PN with respect to the network output representing TDO. Graph dominators provide information about the origin and the end of re-converging paths in a network. A dominator $u \in V$ of a vertex $v \in V$ with respect to some output vertex $w \in V_P$ is a vertex, which is contained in every path starting from v to w. In other words, all data passing a PNs data input also pass the data output of the PN and vice versa. The first efficient algorithm on finding dominators in large graphs has been presented in [23]. In [24] it has been shown that graph dominators can be found in linear time.

The key properties of a PN can be formulated as follows:

1) the data output vertex of a PN dominates the data input vertex with respect to TDO,
2) every component within a PN is reachable from the PN's data input,
3) from every component within a PN there exists a path towards the PN's data output,
4) all ScanMux control bits controlling some multiplexer within a PN control that multiplexer exclusively, and
5) for every possible assignment of values in the MUX-controlling ScanMux control bits, there exists an active scan-path from the PN's data input.

Considering the components depicted in the PN_1-area of Fig. 4 the data output of multiplexer M_1 dominates the data input of PN_1 since every path starting from the PN_1 data input leading to TDO is passing multiplexer M_1. Furthermore the select bit of M_1 is exclusively connected to C_1. Finally since M_1 is a 2-input multiplexer controlled by a single ScanMux control bit, denoted C_1, PN_1 is a perfect network. The rules formulated above are also true for PN_2 and PN_3 in Fig. 4.

In the context of modeling a reconfigurable scan network and computing active scan-paths, the modeling of components within a PN is not required to establish an active scan-path passing through this PN since by construction there always exists a path starting from the data input of the PN reaching the data output of the PN, where every component contained in the path is active.

Furthermore it is possible to derive for every PN the minimum number of scan elements on a scan-path and the corresponding assignment of the relevant ScanMux control bits. This analysis is performed upfront to ease the subsequent reasning process. Due to that all scan segments contained in a perfect network can be removed from the cost function. Hence the minimization problem can be significantly reduced. The derivation of the minimum number of scan elements and the setting of the corresponding control bits is performed independently of the actual re-targeting process.

Please note that SIB-based structures within scan networks fulfill the requirements of a PN by construction. Hence the

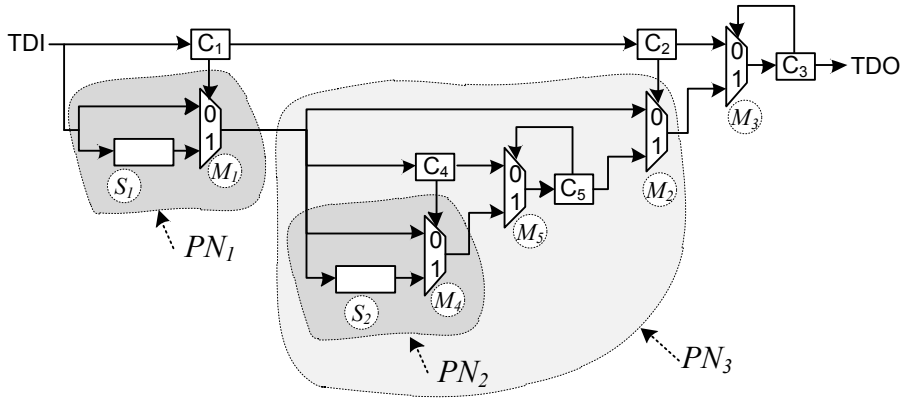

Fig. 4. Perfect network example with three PNs (namely, PN_1, PN_2, and PN_3)

proposed reduction approach can also be applied for SIB-based scan networks.

B. Optimized Reasoning in 1687 Networks

The focus of this subsection is to employ the new upper bound computation described in Section IV and the notion of perfect networks described above in order to propose an efficient and applicable approach to retarget sets of PDL commands in IEEE 1687 networks ensuring the global minimum with respect to the number of CSUs and with respect to the number of shift cycles.

Knowing the upper bound of a sequential problem with respect to the number of time frames enables the immediate computation of a global optimum. In contrast to the approach proposed in [4], where numerous runs of a SAT-solver are required to obtain the minimum of CSUs, our approach only requires a single run of the pseudo-Boolean optimizer MiniSat+ [17]. This is ensured by generating a network representation of the targeted IEEE 1687 network which is unrolled over the number of time frames as determined by the new upper bound computation. The cost function added to the pseudo-Boolean representation is formulated such that every time frame requiring some active scan-path increases the value of the cost function by one. Hence finding a solution with a minimum of activated scan-paths provides the minimum number of CSUs required to execute a set of PDL commands in an IEEE 1687 network.

As described in Section IV this upper bound is also applicable for finding the global minimum with respect ot the number of shift cycles to establish the required assignments within the scan network. In order to obtain the global minimum it is required to replace the cost function for the CSU minimization by a cost function as described in III-B, where it contains all scan elements which are possibly included in some active scan-path during some time frame.

Although the complexity of IEEE 1687 network controls is reasonably small, the sequential problem modeling the IEEE 1687 network over a number of time frames is significantly harder. However, the complexity of the corresponding optimization problems is mostly resulting from the complexity of the cost functions. In other words the solver-internal representation of the cost function is especially for the cycle-based optimization much larger than the representation of the unrolled scan network. The size of the cost function and its solver-internal representation is highly depending on the number of modeled time frames and on the number of considered scan elements. This is due to the larger set of possible numerical solutions. The numerical solution space in the solver is modeled in form of an adder, a sorter, or a BDD-like representation. The applied representation is automatically chosen by a solver-internal heuristic during the reasoning process [17].

PNs reduce the size of the cost function and hence the complexity of its solver internal representation. Due to this reduction, the complexity of the subsequent reasoning process is also significantly decreased. In the following it is described how the concept of PNs is applied to identify redundant elements in the cost function. Consider the scan network depicted in Fig. 4 and let us assume that scan segment S_2 should be activated, then the only path towards TDI passes through PN_1. As discussed in Subsection V-A the minimum length of an active scan-path and the corresponding assignment of the control signals, in this case C_1, are known. Hence in the described case S_1 does not need to appear in the cost function. Only an additional constraint needs to be added to enforce $C_1 = 0$ during the time frame, where S_2 is supposed to be active. Furthermore all constraints ensuring the continuation of an active scan-path towards TDI, as required in the model proposed in [3], are also not required since by construction there always exists a valid path through a PN and PN_1 is directly connected to TDI.

Considering the case that scan segment S_1 in Fig. 4 should be accessed, then every active scan-path starting from S_1 towards TDO is passing PN_3. Again it is possible to apply the above concept, firstly we can rely that there always exists a valid path through a PN and hence the representation of the components within PN_3 is redundant. Secondly it is possible to remove the scan segments, S_2, C_4, C_5, from the cost function. The resulting cost function would only contain elements representing S_1, C_1, C_2 and C_3, in each modeled time frame.

If a set of PDL commands requires access to several instruments or scan segments, then only those scan segments can be removed from the cost function which are not required

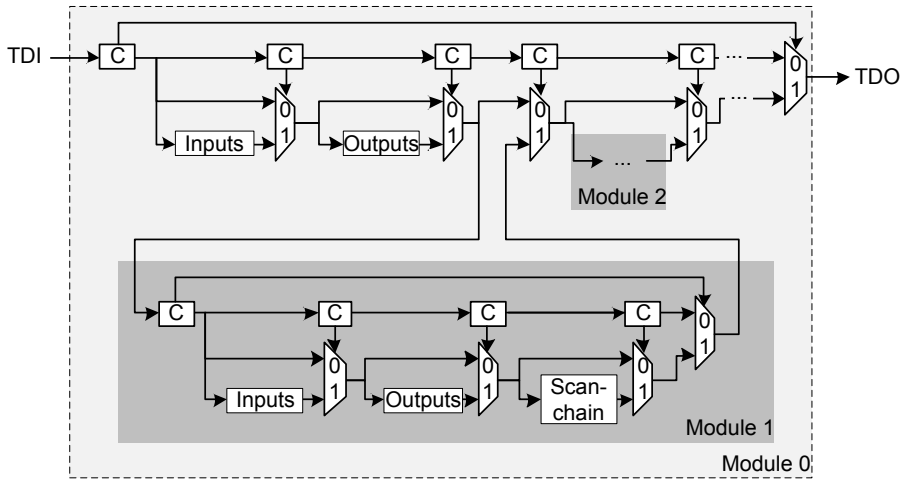

Fig. 5. Part of the network generated for the P34392 benchmark circuit

for any of those instruments or scan segments. Assuming a PDL command block would require access to S_1 and S_2 in Fig. 4, then there are no reductions obtainable by our approach. It is important to emphasize that the reductions achievable by applying PNs depend highly on the set of instruments which need to be accessed. Hence the resulting cost function needs to be derived separately for every set of PDL commands. Futhermore it can be stated that the number and the complexity of applicable PNs is vital to the complexity of the reasoning process since a linear reduction of the size of the cost function leads to an exponential reduction of the solution space.

VI. Experimental Results

The objective of the experiments is to compare the proposed approach in respect to efficiency against the method described in [4]. As mentioned earlier, efficiency is given by the CPU time it takes to generate the scan vectors required to change the network from accessing one set of instruments to accessing another set of instruments. As for the effectiveness, it should be noted that the proposed method is guaranteed to find the optimal solution in terms of access time, due to the use of the upper bound calculation method described in Section IV.

For experimentation, we used a set of designs with networks implemented as the MUX-based architecture described in [4]. These designs are based on the 12 hierarchical circuits in the ITC'02 benchmark set [25]. For each circuit, the number of modules, hierarchical structure, number and type of ports, and number and length of internal scan-chains is available [25]. To create the networks, a number of shift-registers are considered for each module as follows:

- a shift-register with a length equal to the number of input pins,
- a shift-register with a length equal to the number of output pins,
- and one shift-register per internal scan-chain where the length of the shift-register equals that of the scan-chain.

The general architecture of the generated MUX-based networks has the style of the network shown in Fig. 2, where the DFT instrument is placed in the first level of the hierarchy and the Sensor and Debug instruments are placed in the second level. Considering that the ITC'02 benchmark circuits are hierarchical designs, for each benchmark circuit, a corresponding network is designed such that the shift-registers extracted as listed above, are placed in a hierarchical level corresponding to the original hierarchy reported for the ITC'02 circuits. As an example, Fig. 5 shows how a MUX-based network is constructed for the P34392 benchmark circuit. The details of the designs are in the first five columns of Table I listed in columns for design, number of multiplexers, number of scan segments, number of total scan bits, and number of hierarchical levels in the design, respectively.

The modeling approach was implemented in C++. For solving the ILP-problems the pseudo-Boolean solver Minisat+ v1.0 [17] has been used. The experiments were conducted on an INTEL Xeon E5645 2.4GHz with 32GByte main memory running Linux Mint 13 64-bit.

For the experiments, two cost functions are used (discussed in Section V-B): one which finds the minimum number of CSU cycles needed for the retargeting step (i.e., the generated scan vectors use the least number of CSU cycles), and one which finds the minimum access time for the retargeting step (i.e., the generated scan vectors take the least number of test clock cycles for their application). The former cost function is used in [3], while the latter is used in [4].

For every benchmark 10 test cases were applied. Each test case randomly activates 10% of the scan segments in the design. For the benchmarks containing less than 100 scan segments, 10 scan segments were activated. In our experiments the activation of a scan segment S_i implies that the generated sequence of scan vectors, starting from an initial state, ensures that S_i is contained in at least one active scan path established within the test vector sequence.

The results from the experiments are listed in Table I. The benchmarks were translated into pseudo-Boolean (pB) constraints (clauses) and unrolled over a number of time frames determined by the new upper bound computation presented in Section IV. The results of the upper bound computation are reported in the sixth column of the table. The

TABLE I. EXPERIMENTAL RESULTS ON ITC'02 BENCHMARK SET.

Design	Number of muxes	Total scan segm.	Total scan bits	Hierarchical levels	Proposed upper bound (Section IV)	Previous model [4]				Proposed model			
						number of pB clauses	t_{avg}^{csu} [s]	t_{avg}^{cycles} [s]	number of time outs	number of pB clauses	t_{avg}^{csu} [s]	t_{avg}^{cycles} [s]	number of time outs
u226	59	99	1475	2	4	3342	0.060	1.340	0	2030	0.040	0.480	0
d281	67	117	3880	2	4	3816	0.068	1.620	0	2315	0.044	0.820	0
h953	63	109	5649	2	4	3584	0.065	3.066	0	2175	0.044	0.869	0
f2126	45	81	15834	2	4	2580	0.047	1.310	0	1565	0.034	0.460	0
a586710	47	79	41682	3	6	1710	0.060	12.31	0	1040	0.040	1.840	0
q12710	30	51	26188	2	4	1710	0.031	0.970	0	1040	0.020	0.460	0
g1023	94	159	5400	2	4	5322	0.091	196.6	7	3230	0.063	1.320	0
d695	178	335	8407	2	4	10239	0.168	-	10	6195	0.113	4.740	0
p34392	142	245	23261	3	6	9864	0.167	-	10	6002	0.112	11.17	0
t512505	191	319	77037	2	4	10780	0.187	-	10	6542	0.118	3.850	0
p22810	311	565	30139	3	6	21770	0.382	-	10	13230	0.243	96.91	2
p93791	653	1241	98637	3	6	46037	0.810	-	10	27945	0.537	-	10

number of pseudo-Boolean constraints required to represent the unrolled scan network are reported under columns "number of pB clauses" for both previous model (column seven) and the proposed model (column eleven). The reduction of the proposed model compared to previous model in number of pseudo-Boolean constraints for all designs is around 40%. The average run-times to generate scan vectors requiring the minimal number of CSUs for the 10 test cases are listed under columns "t_{avg}^{csu}" for previous model (column eight) and for the proposed model (column twelve). For all designs, the run-times are significantly lower for the proposed model. On average run-times are reduced by 33.4%.

The average run-time to compute minimal scan vector with respect to the number of shift cycles and hence minimal overall access time are listed under columns "t_{avg}^{cycles}" for the previous model (column nine) and the proposed model (column thirteen). The time-out limitation for these experiments was set to 300 seconds. The proposed model computed and proved the minimum of shift cycles within the time-out limit for 108 out of 120 test cases, while the previous model computed and proved the minimum in 63 out of 120 test cases. For several larger benchmarks all test cases timed out using the previous model. Please note that the presented upper bound results were applied to both the previous model and the new model. Hence the listed run-time improvements result only from the reductions in the proposed model.

The effectiveness of the retargeting process has been improved such that due to the proposed upper bound computation the generated scan vector sequences are proved to be minimal.

VII. CONCLUSION AND FUTURE WORK

IEEE 1687 enables flexible access to the embedded (on-chip) instruments. As instruments are to be accessed differently, the IEEE 1687 network will be frequently reconfigured from accessing one set of instruments to accessing a different set of instruments. In this paper, we proposed a scheme based on Boolean Satisfiability Problem (SAT) which uses a pseudo-Boolean optimizer to find the fastest way to perform the reconfigurations. The proposed scheme makes use of our improved modeling and tighter upper bound on the number of capture-shift-update operations to achieve better results compared to previous approaches. The tighter upper bound allows for ensuring optimality while performing only one call to the pseudo-Boolean optimizer, which should be contrasted to previous approaches that made multiple calls to the pseudo-Boolean optimizer with no guarantee on optimality. The improved modeling helps to reduce the size of the models, leading to lower run-times which in turn result in obtaining more optimal solutions than previous modeling concepts—as was highlighted by the presented experimental results.

As future work, the presented upper bound calculation should be developed to be applicable to a wider range of IEEE 1687 networks. Moreover, the presented method for the basic retargeting step can be integrated in a complete retargeting scenario with the aim of achieving minimized total instrument access time.

REFERENCES

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

[2] IEEE Association, "IEEE Std 1149.1-2001, IEEE Standard Test Access Port and Boundary-Scan Architecture," 2001.

[3] R. Baranowski, M. Kochte, and H.-J. Wunderlich, "Modeling, Verification and Pattern Generation for Reconfigurable Scan Networks," in *Proc. IEEE Int'l Test Conf. (ITC)*, 2012, pp. 1–9.

[4] R. Baranowski, M. A. Kochte, and H.-J. Wunderlich, "Scan Pattern Retargeting and Merging with Reduced Access Time," in *Proceedings of IEEE European Test Symposium (ETS'13)*. IEEE Computer Society, 2013, pp. 39–45.

[5] M. Portolan, B. Van Treuren, and S. Goyal, "Executing IJTAG: Are Vectors Enough?" *Design Test, IEEE*, vol. 30, no. 5, pp. 15–25, Oct 2013.

[6] J. Rearick and A. Volz, "A Case Study of Using IEEE P1687 (IJTAG) for High-Speed Serial I/O Characterization and Testing," in *Proc. IEEE Int'l Test Conf. (ITC)*, Oct. 2006, pp. 1–8.

[7] F. G. Zadegan et al., "Reusing and Retargeting On-Chip Instrument Access Procedures in IEEE P1687," *Design & Test of Computers, IEEE*, vol. 29, no. 2, pp. 79 –88, april 2012.

[8] Y. Fkih, P. Vivet, B. Rouzeyre, M.-L. Flottes, G. Di Natale, and J. Schloeffel, "2D to 3D Test Pattern Retargeting Using IEEE P1687 Based 3D DFT Architectures," in *IEEE Computer Society Annual Symposium on VLSI (ISVLSI), 2014*, July 2014, pp. 386–391.

[9] J. P. Marques-Silva and K. A. Sakallah, "GRASP: a search algorithm for propositional satisfiability," *IEEE Transactions on Computers*, vol. 48, no. 5, pp. 506–521, 1999.

[10] M. W. Moskewicz, C. F. Madigan, Y. Zhao, L. Zhang, and S. Malik, "Chaff: Engineering an efficient SAT solver," in *Proceedings of the 38th ACM/IEEE Design Automation Conference*, Las Vegas, Nevada, June 2001, pp. 530–535.

[11] N. Eén and N. Sörensson, "An extensible sat-solver," in *Theory and Applications of Satisfiability Testing, 6th International Conference, SAT 2003. Santa Margherita Ligure, Italy, May 5-8, 2003 Selected Revised Papers*, 2003, pp. 502–518.

[12] N. Eén and A. Biere, "Effective preprocessing in SAT through variable and clause elimination," in *Theory and Applications of Satisfiability Testing, 8th International Conference, SAT 2005, St. Andrews, UK, June 19-23, 2005, Proceedings*, 2005, pp. 61–75.

[13] A. K. J. Baumgartner and J. Abraham, "Property checking via structural analysis," in *Proc. 14 Intl. Conference on Computer Aided Verification (CAV'02)*. Springer Verlag, 2002, pp. 151–165.

[14] K. Yang, K.-T. Cheng, and L.-C. Wang, "Trangen: a SAT-based ATPG for path-oriented transition faults," in *Proceedings of the 2004 Asia and South Pacific Design Automation Conference*, 2004, pp. 92–97.

[15] D. Tille, R. Krenz-Bååth, J. Schloeffel, and R. Drechsler, "Improved circuit-to-CNF transformation for SAT-based ATPG," in *IEEE European Test Symposium, Informal Digest of Papers*, 2008.

[16] C. Liu, A. Kuehlmann, and M. W. Moskewicz, "CAMA: A multi-valued satisfiability solver," in *2003 International Conference on Computer-Aided Design (ICCAD'03), November 9-13, 2003, San Jose, CA, USA*, 2003, pp. 326–333.

[17] N. Eén and N. Sörensson, "Translating pseudo-boolean constraints into SAT," *Journal on Satisfiability, Boolean Modeling and Computation*, vol. 2, no. 1–4, pp. 1–26, 2006.

[18] M. Gebser, B. Kaufmann, A. Neumann, and T. Schaub, "Conflict-driven answer set solving," in *IJCAI 2007, Proceedings of the 20th International Joint Conference on Artificial Intelligence, Hyderabad, India, January 6-12, 2007*, 2007, p. 386.

[19] S. Eggersglüß, K. Schmitz, R. Krenz-Baath, and R. Drechsler, "Optimization-based multiple target test generation for highly compacted test sets," in *19th IEEE European Test Symposium, ETS 2014, Paderborn, Germany, May 26-30, 2014*, 2014, pp. 1–6.

[20] R. Krenz-Baath, A. Glowatz, and F. Hapke, "Fault collapsing of multi-conditional faults," in *16th IEEE International Symposium on Design and Diagnostics of Electronic Circuits & Systems, DDECS 2013, Karlovy Vary, Czech Republic, April 8-10, 2013*, 2013, pp. 42–47.

[21] N. Eén and N. Sörensson, "Temporal induction by incremental SAT solving," vol. 89, no. 4, 2003.

[22] A. Biere, A. Cimatti, E. M. Clarke, and Y. Zhu, "Symbolic model checking without BDDs," in *5th International Conference on Tools and Algorithms for Construction and Analysis of Systems (TACAS'99)*, Amsterdam, The Netherlands, March 1999, pp. 193–207.

[23] T. Lengauer and R. E. Tarjan, "A fast algorithm for finding dominators in a flowgraph," *Transactions on Programming Languages and Systems*, vol. 1, no. 1, pp. 121–141, July 1979.

[24] S. Alstrup, D. Harel, J. Clausen, and M. Thorup, "Dominators in linear time," *SIAM Journal on Computing*, vol. 28, no. 6, pp. 2117–2132, 1999.

[25] E. J. Marinissen et al., "A set of benchmarks for modular testing of SOCs," in *Proc. ITC*, 2002, pp. 519–528.