Power-aware Test Scheduling for IEEE 1687 Networks with Multiple Power Domains

Payam Habiby[*] Sebastian Huhn[*†] Rolf Drechsler[*†]

[*]University of Bremen, Germany
{habiby,huhn,drechsler}@uni-bremen.de

[†]Cyber-Physical Systems, DFKI GmbH
28359 Bremen, Germany

***Abstract*—New test access methodologies are required to cope with the ever-increasing complexity of latest system-on-a-chip designs. The IEEE 1687 standard defines an access methodology to embedded instruments through a reconfigurable scan infrastructure. This technique allows implementing even large networks while keeping the individual access time low since only relevant parts of the scan chain are included in the scan path. However, the reconfiguration introduces a timing overhead, which can be mitigated by accessing instruments concurrently. The concurrent activation of instruments forms a critical aspect from the power management perspective since latest designs consist of multiple power domains with individual power constraints. To avoid any test failures, it must be avoided that the total power consumption of the concurrently activated instruments exceeds the domain's power limit. Particularly when considering highly complex IEEE 1687 networks, which reveal the full potential of the standard by introducing hierarchical and optimized networks, the power-aware test scheduling is a non-trivial task.**

This paper proposes a test scheduling scheme for complex IEEE 1687 networks, which heavily orchestrates graph-based methods. In the end, an optimized test plan is determined, which ensures, on the one hand, a minimized overall test access time and, on the other hand, full compliance with the given power constraints. The approach's efficacy is demonstrated on state-of-the-art benchmark sets involving complex networks with various power domains.

I. Introduction

The next generation of *System-on-a-Chip* (SoC) designs embraces a large number of embedded components ranging from microprocessor cores, digital-signal processors to interface components. An effective access during test to such a variety of heterogeneous on-chip components strictly requires a suitable on-chip infrastructure, as typically introduced by a test network. Such a network can be used for the manufacturing test, fault diagnosis, debug and post-silicon validation as well as for the maintenance, monitoring, and reconfiguration in the field [1]. Typically, the access to this infrastructure is based on scan networks [2]. The IEEE 1687 standard [3] (also known as IJTAG) implements a mechanism to reconfigure the scan chains dynamically. By this, a significant faster access to different segments of the SoC is realized in comparison to the legacy method of daisy chaining [4]. More precisely, such a test network allows the reduction of the instrument access time by varying the length of the scan-path to include only those instruments in the path that are required for the current session. This is realized by introducing components for the reconfiguration like *Scan-mux Control Bits* (SCBs), *Segment Insertion Bits* (SIBs), and *Scan Multiplexers* (ScanMux) [3], as shown in Fig. 1. In particular, SIBs operate as a switch to include or exclude the segments from the chip-level interface. The state of the SCBs and SIBs is defined by a control bit, which is scanned into the corresponding scan registers, and is stored in the update registers by an update signal (from the *Test Access Port* (TAP) controller). Furthermore, Fig. 1 presents the reconfigurable access to the *Test Data Registers* (TDRs) using an IEEE 1149.1 TAP controller [4] over an IJTAG network. These TDRs are registers that are connected to the instruments. In the given exemplary test network, the SIB can exclude TDR1 and TDR2 from the scan path such that the network only includes TDR3. For this purpose, a '0' should be shifted into the SIB's update register such that the multiplexer inside the SIB selects the port '0' and disconnects the upper part of the circuit.

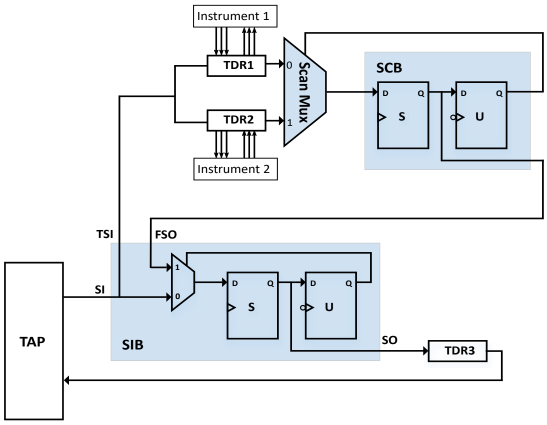

Fig. 1. IEEE 1687 reconfiguration components in an IJTAG network

SoC designs often introduce dedicated modules for the built-in self-test of the memory or the logic itself. However, these modules have highly deviating requirements concerning the test time and energy consumption, which depend on the test type and the component-under-test. As the overall test time has a direct effect on the test cost, obtaining an optimized access sequence to the built-in self-test modules improves the test procedure. On the other hand, using on-chip scan structures creates power issues in non-functional operating conditions, which can violate the power specification. This can in turn

978-1-7281-9457-8/20/$31.00 ©2020 IEEE

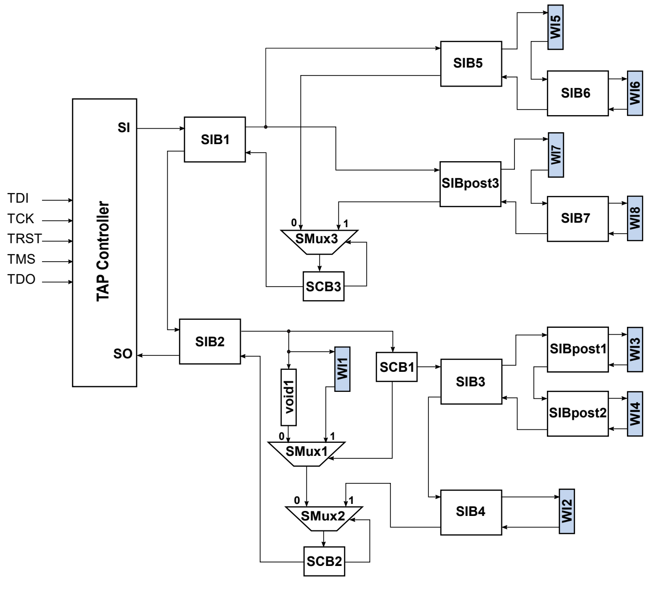

Fig. 2. Mingle network from ITC'16 benchmark suite

lead to unreliable test results [5], [6].

This paper analyzes the instrument access latency over complex IJTAG networks and invokes a test scheduling method, which considers the network's power constraints. By this, an optimized test plan is generated, which allows to reduce the overall test access time significantly. More precisely, this paper proposes a framework to calculate the minimum IJTAG instrument access latency by modeling it as a directed graph shortest path problem. Furthermore, this work discusses in detail how the access latency and the network configuration affect the power considerations. The main contribution concerns the identification of critical read and write paths in a given IJTAG network while considering the resulting latency such that these paths can be avoided in the network design phase. Moreover, the proposed framework investigates the possible concurrency of selected instruments such that none of the given power domain constraints are violated.

The structure of this paper is as follows: At first, a brief overview on prior works about IJTAG is given and the mechanisms of these test networks are briefly presented in Section II. The proposed graph-based model is presented in Section III, which allows representing an IJTAG network while considering multiple power domains. This is followed by a description of the proposed algorithm to calculate the (read/write) latency of IJTAG instruments in Section IV. Furthermore, two different power considerations for the networks are discussed in Section V. Finally, the experimental results are presented in Section VI and the conclusion are drawn in Section VII.

II. Related Work and Background

Several methods have been proposed in [3], [7]–[15] to improve different features of the IJTAG network, including test access time calculation, retargeting access time reduction, securing the IJTAG network and test scheduling. However, some of these works like [10] consider purely SIB-based networks to reduce the modeling complexity. Here, the full potential of IJTAG remains unexploited since components like ScanMuxes are not considered when focusing on these SIB-based networks. In works like [11], [16], only networks are processed that include instruments with dedicated SIBs, which is typically not given within arbitrary networks. Besides this, an algorithm for calculating the overall access time is presented in [9]. The authors in [11] propose an optimization technique to minimize the SIB programming overhead when the instruments are accessed by using a hybrid scheduling scheme. An optimized pattern retargeting is investigated in [12] by projecting it onto a pseudo-Boolean optimization problem and then using an optimization-based *Boolean Satisfiability* (SAT) solving algorithm to determine an optimum scan sequence.

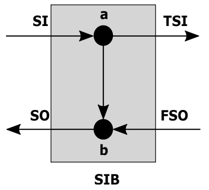

Fig. 3. SIB model for an IJTAG graph representation

This method is improved in [13] by presenting a new bound calculation method to minimize the number of SAT solver invocations. In [14], an on-chip access method, as required for dynamic retargeting, is proposed by mapping the network to a selection dependency graph. A parallel-IJTAG architecture is proposed in [8] to provide higher bandwidth for accessing the instruments by dividing k-bit TDRs into $n(k/n)$-bit TDRs and replacing every single SIB with n SIBs. Parallel testing has been discussed in [15] by proposing a broadcast IJTAG network for accessing replicated copies of embedded cores. The use of scan architectures leads to higher power demand and this makes SoCs more prone to issues like IR-drop and test failures [17]. Therefore, due to the high power consumption of built-in self-test modules during the test mode compared to the functional operation, only a subset of cores can be in execution mode at any time in a power domain. Thus, the remaining components of this power domain have to remain inactive [18]. In [10], [11] optimization methods for test scheduling are presented, which consider power constraints in a network with only one global power domain. However, despite these works which try to design a network from scratch to obtain an optimized access time, this paper investigates a given network to extract the restrictions imposed by its own structure and then performs the test scheduling according to the power constraints. In addition, the model proposed in this paper is applicable to the networks containing instruments with both exclusive and non-exclusive SIBs.

The IJTAG standard [3] introduces two programming languages namely *Instrument Connectivity Language* (ICL) and *Procedural Description Language* (PDL). ICL reflects the intended test network on an abstract level. More precisely, ICL describes only those components that allow access to the instrument through the network and does not include any of the functional logic of the design [3]. The sequence of operations needed for accessing an instrument is defined by

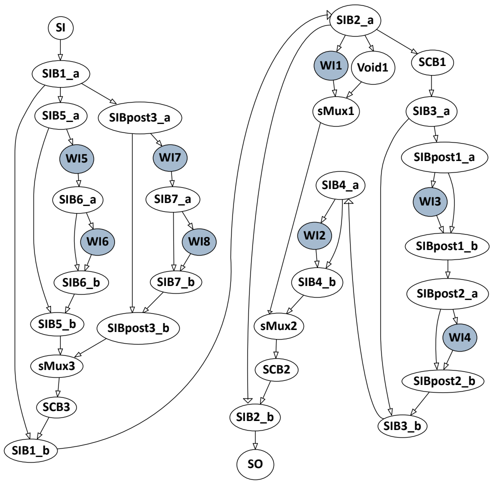

Fig. 4. Equivalent graph of the Mingle network

PDL. In particular, PDL specifies a set of stimuli and expected responses to operate the instruments, which are prepended to any arbitrary test data. At a higher hierarchical level, the instrument level procedures should be translated into the desired level described by ICL. This process of translating to the target module is called retargeting. PDL enables to apply read and write commands simultaneously to different instruments on the IJTAG network, providing that these instruments can already be accessed concurrently via the paths as defined by the ICL netlist. As shown in Fig. 2, the Mingle network from the IEEE 1687 Std. Benchmark suite [19] is an example of an IJTAG network, in which SIBs, SCBs, and ScanMuxes provide reconfigurable access to the instruments.

III. Proposed Network Model

This section describes the proposed model for generic IJTAG network, which contains both SIB as well as ScanMux components. The scan network of Fig. 2 is transformed into a directed graph by parsing the ICL netlist, as shown in Fig. 4. Every SCB and ScanMux is represented by one node and each SIB is modeled with two nodes indexed as 'a' and 'b' as demonstrated in Fig. 3. TDRs are labeled as WI1-WI8. For further clarity of the created graph, only the data path is shown in Fig. 4 and the control signals are ignored. A weight factor is assigned to every vertex according to the number of bits in the register of the instruments that are placed along the data path. Therefore, weights of SIBs and SCBs are 1 and ScanMuxes do not have weight.

Definition 1. *In directed graph $G(V, E)$, a vertex $\nu \in V$ represents a component of the scan network and an edge $e \in E$ is a part of the scan chain that connects two components. Let $w(e)$ be the weight of the edge e and let $p(\nu)$ and $t(\nu)$ be the power consumption and execution time of the node v.*

The shortest path from the ScanIn node to a given vertex v is called write latency and the shortest path from the vertex to the ScanOut node is called read latency. Dijkstra's algorithm [20] is a well-known method for solving the single-source shortest path problem for a graph with non-negative edge weight, which runs in time $O(|E|log|V|)$. Since the time to access a scan segment in a reconfigurable scan network is proportional to the length of the scan path, Dijkstra's method can be employed to calculate the instrument access latency in IJTAG networks. The process starts by assigning a 0 distance to the source vertex and labeling it as visited[1]. In the next step, the distance of the neighbor nodes from the currently visited node is updated if the new distance is less than their initial distance. When all the neighbors are visited, the closest unvisited vertex is added to the shortest path and is selected for the next visit. This procedure continues until the shortest paths to all vertices are obtained.

IV. Latency Calculation

To obtain the minimum access latency for an instrument in the IJTAG network graph, it is assumed that the network has access to all hierarchies. Dijkstra's algorithm operates on the graphs with weighted edges. However, since any vertex cost function can be expressed as a constrained edge cost function, we can apply it on our graph, in which the vertex cost function $f(v) > 0$ is defined as the size of instrument's registers. As proved in [21], an arbitrary cost function $f(v) > 0$ is equivalent to a cost function $g(u, v) > 0$ defined over the set of edges, in which the cost of each edge is the average of the costs of the vertices it connects:

$$g(u, v) = \frac{f(u) + f(v)}{2}$$

Therefore, if we define graph $G = (V, E)$ with weighted vertices $u \in V, v \in V$ as follows:

$$E = \{e | e = uv, w_{\mathrm{u}} = weight(u), w_{\mathrm{v}} = weight(v), \\ w_{\mathrm{e}} = weight(e) = 0\}$$

The modified graph $G' = (V', E')$ with unweighted vertices $u' \in V', v' \in V'$ and weighted edges can be solved using Dijkstra's algorithm with the complexity $O(|E| + |V|log|V|)$:

$$E' = \{e' | e' = u'v', u' = u, v' = v, w_{\mathrm{u}}' = w_{\mathrm{v}}' = 0, \\ w_{\mathrm{e}}' = \frac{(w_{\mathrm{u}} + w_{\mathrm{v}})}{2}\}$$

Accessing an instrument in an IJTAG network, i.e., writing data to this instrument, first requires to establish a path from the scan-in port of the top module to that instrument. However, for the read cycle, the path starts from the instrument and ends at the main scan-out port. Thus, the shortest path for both parts of the scan-chain has to be calculated to reflect a complete read-write cycle. Since Dijkstra's solution provides the shortest paths from one vertex to all other vertices, the calculation has to be conducted for two scan-in and scan-out nodes only, which allows obtaining the shortest read-write chain for every instrument. This prevents two runs of Dijkstra's algorithm for every instrument in the network, saving run-time. The only difference is that the algorithm from the scan-out port should be applied to the graph after reversing the direction of all edges. Algorithm 1 describes the process of finding the shortest chain

[1] Note that the initial distance to other nodes would be infinity that have not been visited yet.

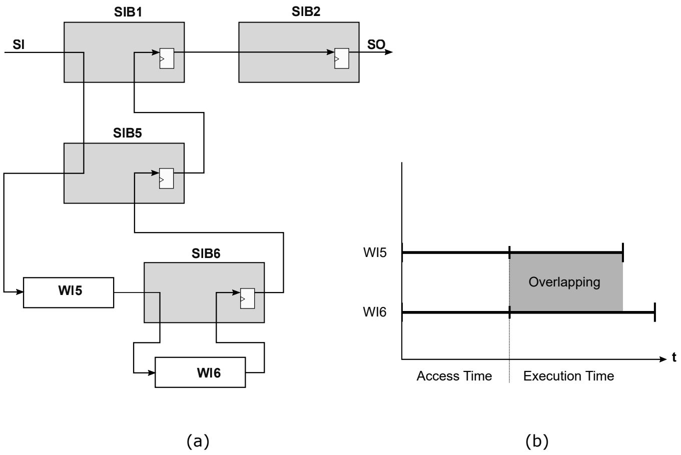

Fig. 5. Concurrent access to WI5 and WI6 in mingle benchmark network (a) an the execution time overlapping (b).

```
Algorithm 1: Determining the shortest chain in IJTAG
    Input: $ICL, maximumLatency$
    Output: $read/writeLatency, shortestChain$
    $G(V, E) = Parse(ICL)$;
    function ModifyGraph($G$):
        foreach ($e = uv \in E$) do
            $w_{e'} = \frac{w_u + w_v}{2}$;
            $w_{u'} = w_{v'} = 0$;
        end
        $G = (V', E')$;
    end
    $G'$ =ModifyGraph($G$);
    $shortestWritePath$=Dijkstra$_{\text{from scan-in}}(G')$;
    $G''$=ReverseEdges($G'$);
    $shortestReadPath''$ = Dijkstra$_{\text{from scan-out}}(G'')$;
    $shortestReadPath$ =Reverse($shortestReadPath''$);
    foreach ($v \in V$) do
        $writeLatency = \sum_{shortestWritePath} registers$;
        $readLatency = \sum_{shortestReadPath} registers$;
        if $readLatency > maximumLatency \vee$
          $writeLatency > maximumLatency$ then
            print "Latency Violation";
        else
            $shortestChain =$
              $shortestWritePath + shortestReadPath$;
        end
    end
end
```

including a given instrument and checking the path for possible latency violations.

V. Power Analysis

With the increase in the number of IP blocks in SoC designs, power consumption has emerged as a major design constraint. Therefore, power management and energy awareness considering multiple power domains have gained great importance in the latest SoC designs. Therefore, besides the read and write latency analysis, the power constraint has to be further investigated. This issue is discussed for both, the concurrent as well as sequential test scheduling in Section V-A and Section V-B, respectively.

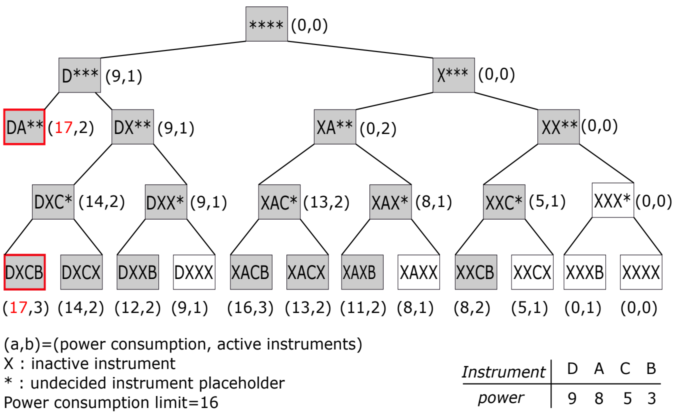

Fig. 6. Checking the power constraint for concurrent execution of four instruments: *A, B, C, D*

A. Concurrent Access

The instruments in an IJTAG network can be accessed either concurrently or sequentially. In concurrent access, the required data for instruments are concatenated before applying to the network and then shifted into the TDRs of the established chain. Therefore, access to all instruments in the chain occurs at the same time. Consequently, the instruments start the execution phase simultaneously. For instance, in Fig. 5a, the chain is configured during the concurrent access such that it includes the concurrent TDRs. Fig. 5b shows that *WI5* and *WI6* overlap in execution phase as soon as their connected components start running. The investigation of the power constraints during this access method is about checking whether the sum of power consumption of concurrently running instruments exceeds the maximum allowed power limit. All possible combinations of the instruments have to be checked to determine the maximum allowed concurrent nodes. The overall number of possible activated instruments would be $\sum_{i=0}^{n} C(n,i)$. Fig. 6 depicts how the dynamic analysis contributes (a) to the search space reduction and (b) to find power constraint violation in a scan-chain consisting of four instruments. In each branch division of the tree (as given in Fig. 6), the inclusion or the exclusion of one component is investigated. The asterisk symbols (*) indicate that no decision has been made yet. The X marks indicate that the related instrument is excluded and should not be in execution mode. For every box, the total power of the included components would be calculated. In the case of any violations as in *"DA**"*, the algorithm stops investigating the sub-branches and issues a violation warning. In the last level of the tree, it can be seen that four instruments can run concurrently in six different combinations. It should be noted that white boxes, which include less than two instruments, are not required to be checked, given that all instruments already comply with the power consumption requirements. Therefore, these boxes cannot exceed the power limit.

Concurrent access to instruments is not always possible in the IJTAG network. For instance, no chain can be found to include both *WI5* and *WI7*, as stated in Fig. 2. Thus, the first step before investigating the power requirements is about checking the possibility of concurrency. As shown in Fig. 4,

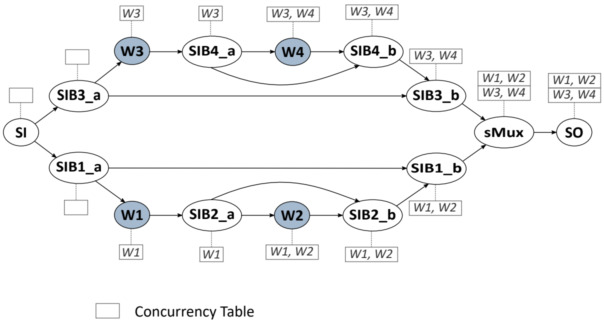

Fig. 7. Identifying concurent groups in an IJTAG network

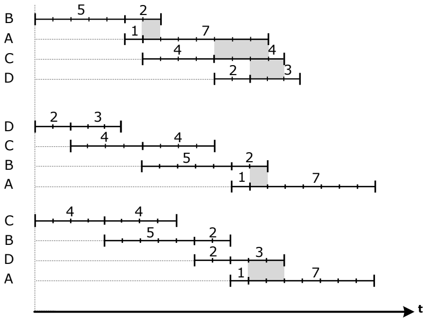

Fig. 8. Effect of the access sequence on the execution time

the IJTAG network constitutes an acyclic directed graph. Every node in this network is located on a path that starts from TDI and ends at TDO. As a result, the number of the concurrent chains is equal to the number of paths from TDI to TDO. In other words, the set of instruments on such paths are suitable to be included in one access chain. Otherwise, the concurrent access to the instruments is not possible and the sequential access should be considered, as stated in Section V-B. Fig. 7 depicts a graph representation of an IJTAG network, which contains four instruments labeled as W1-W4. Every node is assigned a table with all possible concurrent instruments that can be visited through all traversals from SI to that node. As long as a new instrument is not visited, the concurrency table is empty. Every node inherits the previous node's data, and when a new instrument is visited, it is added to the table. For an exemplary vertex SIB2_b, which inherits the {W1} and {W1, W2} tables via two different paths, {W1, W2} as the super-set is assigned to SIB2_b. In a node like sMux, which receives two different {W1, W2} and {W3, W4} tables, these sets are merged as {W1, W2 | W3, W4}.
The traversal is performed according to *Breadth-First-Search* (BFS) algorithm. As the BFS is running, vertices are divided into three groups (1) closed nodes that are visited through all their inputs and therefor their table will not be further updated, (2) open nodes which are not visited yet and their tables are empty and (3) semi-open nodes which are visited via part of their inputs; the tables of these nodes have not received their final value. BFS would not proceed to the next stage unless all tables of the current stage are updated with their final values. At the beginning of the next stage, all concurrency data of these tables are transferred to the new semi-open nodes and the previous tables are deleted. The concurrency table at SO includes all instruments while providing the maximum possible concurrency from the network structure perspective. Every access scheduling scheme should consider the concurrency groups to avoid any adverse impact on the scan chain. The algorithm described in Fig. 6 allows investigating whether these chains comply with the power constraints. After determining the power compatible chains, a certain configuration priority has to be set. For instance, in the aforementioned example, the priority of {W1, W2} and {W3, W4} should be defined. The sequence of configuring these chains possibly affects both test time and energy considerations, as described in the next section in more detail.

B. Sequential Access

In cases where concurrent access is not possible, instruments are accessed sequentially. In other words, the IJTAG network provides serial access to the instruments, i.e., shifting the data to and from a TDR can be started only when access to the previously scheduled instruments is finished. Even in this circumstance, it is still likely that sequentially accessed instruments have an overlapping execution time. For instance, in Fig. 7 while the instruments on the chain *SI-WI1-WI2-SO* are running, a new configuration of the network can provide access to *SI-WI3-WI4-SO*. However, the feasibility of the execution of instruments on this chain depends on their power consumption. When accessing the instruments on a chain, if the execution of the instruments on the previously scheduled chain has been finished, the instruments do not overlap. Consequently, these instruments do not contribute to exceed the domain's maximum power limit. However, if a set of nodes is accessed while the nodes accessed by the previous chain are still under execution, the power constraint has to be checked to ensure that the power requirements are met. Fig. 8 shows the sequential access to four chains in three different sequential orders. Each vector is divided into two parts. The left part shows the access time and the right part represents the maximum execution time of the instruments on a chain. The dim areas present the cases, in which the instruments of two chains are running simultaneously. Changing the order of access directly affects the execution time overlapping and, consequently, alters the concurrent power consumption. Given n chains, the total number of permutations is $n!$, which has been considered to determine the least overlapping candidate. However, as long as the overlapping does not cause a power violation, it is possible to assume a related order. Fig. 8 demonstrates that the overall test time varies only according to the execution time of the last chain in the sequence and, hence, accessing the chain that has the minimum execution time after other chains reduces the overall test time significantly.

VI. Experimental Results

This section presents the experimental results when applying the proposed scheme to the Mingle, BasicSCB, TreeFlat, q12710, N17D3, N32D6, and H953 networks from the ITC'16 benchmark set [19]. The overall framework is written in C++ and all experiments have been conducted on a machine holding an *Intel Core i7-8565U* processor and 16GB of main

TABLE I
BENCHMARK RESULTS

Network		**#(Nodes,Edges)**	**#Inst.**	**#Conc. Groups**	**Max Write Latency**		**Min Write Latency**		**Max Read Latency**		**Min Read Latency**	
Cmplx.	Name				Clk	Inst. Name	Clk	Inst. Name	Clk	Inst. Name	Clk	Inst. Name
B	Mingle	(35,39)	8	4	51	WI6, WI8	18	WI5, WI7	24	WI3	18	WI1
	BasicSCB	(31,42)	5	2	19	WI4, WI5	16	WI1	20	WI2	18	WI5
	TreeFlat	(48,61)	11	1	26	WI_10	4	WI_0	27	WI_0	5	WI_10
	H953	(149,150)	47	1	194	m1::wrpOut	8	wrpIn	234	m1::wrpIn	22	m6::wrpOut
C	q12710	(77,78)	23	1	1,900	m2::wrpIn	221	m1::sc3	1,908	m2::wrpIn	13	m2::sc1
A	N17D3	(57,66)	27	256	189	R21	5	R16A	103	R11t	7	R5B
	N32D6	(90,114)	44	1024	20030	R22B	834	R31	16973	R15B	55	R0A

memory. The considered benchmark networks are divided in three different complexity classes as follows: Basic (B), Classic (C) and Advanced (A). Table I presents the number of nodes, edges and instances in the graph that is created from the aforementioned networks. The maximum and minimum latency are further calculated in clock cycles for writing and reading the data to and from the instruments inside these networks. The paths to the instruments that have the maximum read or write latency are considered as critical paths, therefore, placing an instrument that requires a large number of read or write accesses would result in higher test time and should be avoided. Some names of the instruments in the Table I, like the m2::wrpIn, include the information of the hierarchy of the component in the ICL netlist. In this case, wrpIn represents an instrument inside the module M2, which is instantiated under the name m2 in the top module. The number of concurrent paths when - all SIBs are opened - is shown in the fourth column of Table I. It can be noted that q12710, TreeFlat and H953 have one concurrent path, which includes all instruments of the network. In contrast to this, the N32D6 network holds a large number of 1,024 concurrent paths. The results show that the presented graph model as proposed by this paper, is exploited successfully to calculate the access latency of instruments in all three complexity groups of IJTAG network benchmarks. Furthermore, the proposed method extracts possible concurrent paths in these networks, which provides more accurate overview of IJTAG network for scheduling purposes.

VII. CONCLUSION

This paper proposed a novel test scheduling technique for complex IJTAG test networks, which considers for the first-time multiple power domains. In particular, the main contributions of this paper were as follows: At first, the deliberate network analysis to identify the chains that can include instrument, which are executed in a concurrent fashion, and, secondly, the power consumption analysis over the identified chains (with the concurrently executed instruments) to ensure compliance with the given power restriction of the individual domain. In the end, the proposed technique determined an optimized test plan, which was fully compliant with the given power restriction on the individual power domain. The proposed technique ensured to avoid any unintended test failures, for instance, due to IR-drop, when determining an optimized test plan.

VIII. ACKNOWLEDGMENT

This work was supported by the AI initiative of the Free Hanseatic City of Bremen. The authors gratefully thank Stephan Eggersglüß from Mentor Graphics (Germany) for the inspiring research discussions and thoughtful feedback.

REFERENCES

[1] R. Baranowski, M. A. Kochte, and H.-J. Wunderlich, "Reconfigurable scan networks: Modeling, verification, and optimal pattern generation," *ACM Transactions on Design Automation of Electronic Systems (TODAES)*, vol. 20, no. 2, 2015.

[2] A. Damljanovic, A. Jutman, G. Squillero, and A. Tsertov, "Post-silicon validation of IEEE 1687 reconfigurable scan networks," in *IEEE European Test Symposium (ETS)*, 2019, pp. 1–6.

[3] "IEEE standard for access and control of instrumentation embedded within a semiconductor device," *IEEE Std 1687-2014*, pp. 1–283, 2014.

[4] "IEEE standard for test access port and boundary-scan architecture," *IEEE Std 1149.1-2013*, pp. 1–444, 2013.

[5] H. Dhotre, S. Eggersglüß, K. Chakrabarty, and R. Drechsler, "Machine learning-based prediction of test power," in *IEEE European Test Symposium (ETS)*, 2019, pp. 1–6.

[6] P. Girard, N. Nicolici, and X. Wen, *Power-aware testing and test strategies for low power devices*. Springer Science & Business Media, 2010.

[7] M. A. Kochte, R. Baranowski, and H. Wunderlich, "Trustworthy reconfigurable access to on-chip infrastructure," in *IEEE International Test Conference in Asia (ITC-Asia)*, 2017, pp. 119–124.

[8] S. Gupta, A. Crouch, J. Dworak, and D. Engels, "Increasing ijtag bandwidth and managing security through parallel locking-sibs," in *IEEE International Test Conference (ITC)*, 2017, pp. 1–10.

[9] F. G. Zadegan, U. Ingelsson, G. Carlsson, and E. Larsson, "Access time analysis for IEEE p1687," *IEEE Transactions on Computers (TOC)*, vol. 61, no. 10, pp. 1459–1472, 2012.

[10] F. G. Zadegan, U. Ingelsson, G. Asani, G. Carlsson, and E. Larsson, "Test scheduling in an IEEE p1687 environment with resource and power constraints," in *IEEE Asian Test Symposium (ATS)*, 2011, pp. 525–531.

[11] S. S. Nuthakki, R. Karmakar, S. Chattopadhyay, and K. Chakrabarty, "Optimization of the IEEE 1687 access network for hybrid access schedules," in *IEEE VLSI Test Symposium (VTS)*, 2016, pp. 1–6.

[12] R. Baranowski, M. A. Kochte, and H. Wunderlich, "Scan pattern retargeting and merging with reduced access time," in *IEEE European Test Symposium (ETS)*, 2013, pp. 1–7.

[13] R. Krenz-Baath, F. G. Zadegan, and E. Larsson, "Access time minimization in IEEE 1687 networks," in *IEEE International Test Conference (ITC)*, 2015, pp. 1–10.

[14] A. Ibrahim and H. G. Kerkhoff, "Analysis and design of an on-chip retargeting engine for IEEE 1687 networks," in *IEEE European Test Symposium (ETS)*, 2016, pp. 1–6.

[15] S. Gupta, J. Wu, and J. Dworak, "Efficient parallel testing: A configurable and scalable broadcast network design using IJTAG," in *IEEE VLSI Test Symposium (VTS)*, 2018, pp. 1–6.

[16] F. G. Zadegan, U. Ingelsson, G. Carlsson, and E. Larsson, "Design automation for IEEE p1687," in *Design, Automation and Test in Europe Conference (DATE)*, 2011, pp. 1–6.

[17] H. Dhotre, S. Eggersglüß, and R. Drechsler, "Cluster-based localization of ir-drop in test application considering parasitic elements," in *IEEE Latin American Test Symposium (LATS)*, 2019, pp. 1–4.

[18] A. M. Rahmani, M. Haghbayan, A. Miele, P. Liljeberg, A. Jantsch, and H. Tenhunen, "Reliability-aware runtime power management for many-core systems in the dark silicon era," *IEEE Transactions on Very Large Scale Integration Systems (TVLSI)*, vol. 25, no. 2, pp. 427–440, 2017.

[19] "IEEE std. 1687 benchmark networks," 2016. [Online]. Available: https://gitlab.com/IJTAG/benchmarks

[20] E. W. Dijkstra *et al.*, "A note on two problems in connexion with graphs," *Numerische mathematik*, vol. 1, no. 1, pp. 269–271, 1959.

[21] M. Barbehenn, "A note on the complexity of Dijkstra's algorithm for graphs with weighted vertices," *IEEE Transactions on Computers (TOC)*, vol. 47, no. 2, pp. 263–, 1998.