Optimal Power-Constrained SoC Test Schedules With Customizable Clock Rates

Vijay Sheshadri, Vishwani D. Agrawal and Prathima Agrawal
Department of Electrical and Computer Engineering, Auburn University, Auburn, AL 36849
Email: {vijay, agrawvd, agrawpr}@auburn.edu

***Abstract*—In this paper, we propose a method of minimizing test time in SoCs (System-on-chip), for a given power budget, by varying the test clock frequency for each test session. Since frequency is proportional to the test time and the power dissipated, by controlling the test clock frequency, the power dissipated and the test time per session can be adjusted so as to yield an optimal solution to the test scheduling problem. To achieve this, we modify the existing ILP (Integer-Linear Program) model for optimal test scheduling to include a variable frequency parameter which, in turn, controls the test time and power. For the optimization, we have used an open-source ILP solver. We also prove that the lower bound on the total test time of an SoC, is obtained by executing individual cores (tests) per session at their maximum frequency of operation, such that their test power is same as the power budget. Results show an improvement of 27% over existing solution for the benchmark SoC, ASIC Z.**

I. Introduction

Advances in technology have made it possible to integrate an entire system onto a single chip. Owing to their modularity, small area and low power consumption, SoC (System-on-Chip) devices are becoming increasingly popular. Not only are the cores in increasing in number, but also in complexity. As a result of this, effective and efficient testing of SoC poses a challenging problem. Researchers, in the past, have taken different approaches to tackle this problem. Some of the approaches involve designing the test architecture [1], [2], optimizing test schedule [3]–[5], efficient designing of wrapper and test access mechanism (TAM) [6]–[8].

The SoC testing problem can be modelled as a 3-Dimensional optimization problem, where the SoC's power limit, test time and resources (such as pin count, etc.) form the three axes. The power limit is fixed for the SoC and the resources have a limited availability. The objective of the 3-D optimization would be to minimize the test time by effective allocation of resources such that the power limit is not exceeded. This optimization problem has been modelled as a 3-D bin packing problem in [9], as shown in Figure 1. Each core in the SoC can be modelled as a cuboid, where the core's test power, test time and test resources, such as BIST resources, wrapper width etc. constitute the three dimensions. The idea here is to place the cores in the cuboid representing the SoC in such a way that the test time is minimized while satisfying the power and resource constraints. As the number of cores and their complexity increases, test data and time required to test the SoC also increases. While testing multiple cores simultaneously can reduce the test time, such concurrent execution is limited by power consumption during the test and resource availability. A number of power-aware test strategies have been developed in the past [10], one of them being power-constrained test scheduling [7], [11]–[13], where multiple tests are scheduled to run simultaneously in sessions in such a way that the test time is minimized while considering test power constraints and test resource conflicts. The existing tests scheduling techniques can be classified broadly into:

- Non-partitioned testing, where the no new tests are allowed to start until all the tests in the session are completed [14], [15].
- Partitioned testing with run to completion, where the idea of a test session is ignored and a test may be scheduled to start as soon as possible and run until it completes [16].
- Partitioned or pre-emptive testing, which is similar to the previous strategy but the tests can be interrupted at any time. However, all tests must be completed by the end of testing [12].

Some of the earlier work formulates the test time as a function of TAM width and optimize TAM allocation among cores, to achieve test time minimization [8]. The TAM is responsible for transporting the test vectors to the cores and the test response from the cores. In [17], it was shown that, for a given core, the test time varies, as a staircase function, with the TAM width. [5], [7], [8], [13] have shown that, by efficient assignment of TAM width, test time optimization can be achieved.

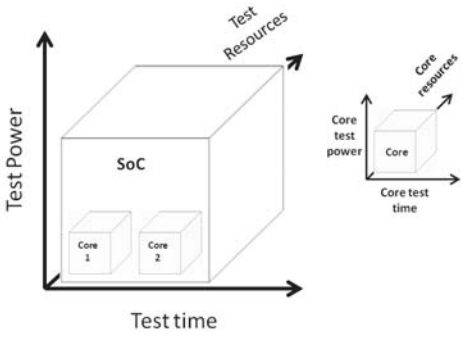


Fig. 1. SoC Test scheduling modelled as 3D optimization problem

In this paper, we formulate the test time and test power as a function of the test clock frequency and achieve test

time minimization, for a predefined power constraint, by setting the clock frequency for each test session. The idea of varying clock speed to reduce test time has been proposed earlier in [18] for ASICs. While [18] adopts a dynamic clock control based on the circuit's activity factor, in our case the clock frequency is dependant on the power dissipation of the test session. The remainder of the paper is organized as follows: Section II provides the mathematical formulation of the problem. In Section III, we propose our approach to obtain test minimization, while the application of this method is provided, in the form of a case study, in Section IV. Section V concludes the paper.

II. Problem Statement

In this section, we describe the proposed test scheduling method in the form of an Integer Linear-Programming model. We define the power budget for an SoC, P_{max}, as the maximum allowable power dissipation during the testing of SoC. Let there be *n* cores, C_1,...,C_n in an SoC and let the test corresponding to a core C_i be t_i, where $i \in 1, 2, .., n$. Each test is associated with test time (referred to as test length throughout this paper) and test power. Let L_{t_i} and P_{t_i} be the length and power of the test t_i. Let the tests, t_1,...,t_n, be distributed among *k* sessions, S_1,...,S_k such that each session, S_j contains one or more tests. The test length of a session S_j, given by $L_{S_j} = \max(L_{t_i} | \forall t_i \in S_j)$ and the power dissipated during session, S_j is given by $P_{S_j} = \sum(P_{t_i}), \forall t_i \in S_j$.

The classic test scheduling problem can be formulated as an ILP model (*M1*).

Objective: Minimize $\sum_{j=1}^{k} L_{S_j}.x_j$, where

$$x_j = \begin{cases} 1, & \text{if } S_j \text{ is scheduled} \\ 0, & \text{otherwise} \end{cases}$$

Subject to: 1) $P_{S_j}.x_j \leq P_{max}$, P_{max} being the power budget for the SoC. 2) each test, $t_i, i \in \{1, 2, .., n\}$ is executed at least once.

Since the classic test scheduling problem does not, explicitly, specify a frequency of operation, let us assume that all test sessions are executed at a nominal test clock frequency,f_{nom} and that the test length and power for each test are characterized at this nominal frequency. As frequency is directly proportional to the power consumed during a test and inversely proportional to the test time, increasing the frequency of the test clock, in turn, increases the test power and lowers the test time. On the other hand, decreasing the frequency results in a reduction in the power and an increase in the test time. Based on this observation, in this work, we modify the classic test scheduling problem by including a new term, referred to as frequency factor, in the objective function. The frequency factor of a session, S_j is defined as, $F_j = \frac{f(S_j)}{f_{nom}}$ where $f(S_j)$ is the frequency of operation of session, S_j. A frequency factor greater than 1 for a session implies that the test clock frequency is being increased for that session, while a frequency factor less than 1 implies that the that the test clock frequency is being decreased for that session. Now, the test scheduling problem can be formulated as an ILP model (*M2*) as follows:

Objective: Minimize $\sum_{j=1}^{k} (L_{S_j}/F_j).x_j$

Subject to: 1) $P_{S_j}.F_j.x_j \leq P_{max}$, where P_{max} is the power budget for the SoC. 2) each test, $t_i, i \in \{1, .., n\}$ is executed at least once.

The first constraint implies that the maximum value to which F_j can be increased, without exceeding the power budget P_{max}, is (P_{max}/P_{S_j}). Here, it is assumed that the increase in the frequency of operation is limited only by the power constraint and that the power of individual tests (cores) can be increased to equal the power budget, P_{max}. If all the sessions execute at the nominal frequency, then $f(S_j) = f_{nom}$ which implies that $F_j = 1$. Substituting this value into the ILP model *M2* transforms it into the ILP model *M1*, which is the classic test scheduling problem. Therefore, we can state that the classic test scheduling problem is a special case of the test scheduling problem formulated as the ILP model *M2*.

III. Proposed Method

Once the mathematical model has been established, the next step is to incorporate this model into the test scheduling process. The resource sharing graph of an SoC provides complete information about the test sets for the cores and the testing resources required by these tests. The resource graph is represented by a bipartite graph where the edges connect tests with their corresponding resources. A resource conflict arises when edges from multiple tests connect to the same resource. Such tests cannot be scheduled for concurrent execution. A test compatibility graph (TCG), as shown in Figure 2, can be constructed from the bipartite graph [14]. In the TCG, an edge between nodes implies that the tests, represented by the nodes, can be scheduled in the same session for concurrent execution.

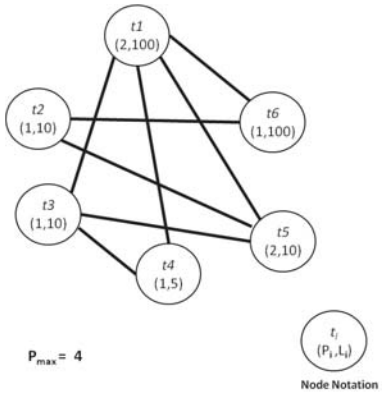


Fig. 2. An example Test Compatibility Graph (TCG) [14]. The test time and test power are in arbitrary units.

From the TCG, we derive all the cliques (maximal subgraphs) of the graph using the well-known Bron-Kerbosch algorithm [19], whose worst-case running time is O($3^{n/3}$) for any *n*-vertex graph. For our purposes, we adopt a python implementation

TABLE I
ALL POSSIBLE TEST SESSIONS DERIVED FROM THE TCG IN FIGURE 2.

Session	Tests	Test Length	Power
1	T1,T3,T5	100	5
2	T1,T3,T4	100	4
3	T1,T6	100	3
4	T1,T5	100	4
5	T1,T4	100	3
6	T1,T3	100	3
7	T2,T6	100	2
8	T2,T5	10	3
9	T3,T5	10	3
10	T3,T4	10	2
11	T1	100	2
12	T2	10	1
13	T3	10	1
14	T4	5	1
15	T5	10	2
16	T6	100	1

of this algorithm [20]. Next, we list all the sub-cliques within each clique. This is done simply by considering each clique and listing all the node combinations possible in the clique; repeated node combinations are deleted at the end. The cliques and the sub-cliques together form a complete set of test sessions possible for a given TCG, as shown in Table I. A subset of these sessions is selected such that the total test time is optimized while satisfying the constraints mentioned in the ILP models *M1* and *M2*. Such a selection can be achieved by solving the ILP models through ILP and various other optimization techniques [3]–[5]. The optimal test time obtained for the TCG in Figure 2, when solved as a classic test scheduling problem (ILP model *M1*) is 120 (Table II).

While the classic test scheduling problem is linear, in the proposed model *M2*, the objective function and one of the constraints are non-linear due to the inclusion of the frequency factor as a variable. However, we linearise the non-linear model by substituting the frequency factor of each session by its maximum value and then solving the now linear model using an ILP solver. We have used GLPK (GNU Linear Programming Kit) solver [21] to perform optimizations through Integer Linear-Programming. The result of including the frequency factor is seen in Table II. The new optimal test time is now 86.25, which is better than the previous optimal test schedule by 28%. It also interesting to note that each session contains only one test and, while the power consumed per session is P_{max}, the test length of each session is $(L_{t_i}.P_{t_i})/P_{max}$. This optimal schedule is the lower bound for the optimal test time of the system represented by the TCG in Figure 2. It can be proved that the lower bound for the optimal test time of an SoC is achieved when, in each session, one individual test is executed and the frequency of operation during the session is $(P_{max}/P_{t_i}).f_{nom}$, where t_i is the only test scheduled in that session. The proof is as follows:
Theorem: Executing one test per session, at a clock frequency such that the power consumed per session is the same as the power budget (P_{max}), gives the smallest total test time and hence forms the lower bound for the optimal test time of the given SoC.
Proof: Let there be *n* tests, t_1,...,t_n. Let L_i and P_i be the test, t_i's length and power dissipated, respectively. Let P_{max} be the power budget.
Case 1: One test executed per session.
Let each test, t_i be executed in session S_i. This implies that the session S_i's length and power are the same as that of the test, t_i. Now, if the clock frequency per session is increased, for speeding up the testing, the frequency factor, $f_i = P_{max}/P_i$. The modified session length, now, is = $L_i/f_i = L_i.P_i/P_{max}$. The total test time (say L1), therefore, is = $\sum_{i=1}^{n} L_i.P_i/P_{max}$ or,

$$L1 = (L_1.P_1 + ... + L_n.P_n)/P_{max} \quad (1)$$

Case 2: Tests executed in multiple sessions.
Let the *n* tests be scheduled in *k* sessions. The length and power of a session, S_j are given by $\max\{L_{ij}\}$ and $\sum(P_{ij}), \forall i \in j$, respectively. If the clock frequency per session is increased, for speeding up the testing, the frequency factor per session, $f_j = P_{max}/\sum P_i, \forall i \in j$. The modified session length for session, S_j is given by, $(\max\{L_{ij}\}.\sum P_{ij})/P_{max}, \forall i \in j$. Let $\max\{L_{ij}\} = L_{max_j}$. The test length of session, S_j, now, is $(L_{max_j}.\sum P_i)/P_{max}, \forall i \in j$. The total test time (say L2), therefore, is = $[\sum_{j=1}^{k} L_{max_j}(\sum P_i)]/P_{max}$, $\forall i \in j$ or,

$$L2 = [L_{max_1}.(P_1+...+P_x)+...+L_{max_k}.(P_y+...+P_n)]/P_{max} \quad (2)$$

where $x, y \in \{1, .., n\}$.
For any session, S_j, $L_{max_j} \geq L_{ij}, \forall i \in j$. This implies that $L_{max_j}.P_{xj} + ... + L_{max_j}.P_{yj} \geq L_{xj}.P_{xj} + ... + L_{yj}.P_{yj}, \forall x, y \in j$. From this and eqn. (1) and (2), we can say that L2 $\geq$ L1 and hence, L1, being the smallest total test time, forms the lower bound.

TABLE II
OPTIMAL SCHEDULES OBTAINED FOR THE TCG IN FIGURE 2 BY SOLVING ILP MODELS.

ILP model *M1*		ILP model *M2*		
Session	**Test Length**	**Session**	**Freq. factor**	**Test Length**
T1,T6	100	T1	2	50
T2,T5	10	T2	4	2.5
T3,T4	10	T3	4	2.5
		T4	4	1.25
		T5	2	5
		T6	4	25
Total test time =	120	Total test time =		86.25

Here, the clock frequency of the cores was limited only by the power budget of the SoC. However, in reality, the frequency of operation of individual cores is often limited by structural constraints (for e.g., critical path delay) and the maximum

power limit of the cores. Taking this into account, let F_m be the upper limit to which the frequency factor of any session can be increased. This implies, $0 \leq F_j \leq F_m$, i.e., the frequency factor of each session S_j can take any value within the range [0, F_m] as long as the power consumed during the session does not exceed P_{max}. For the classic test scheduling problem described by ILP model *M1*, the range for the frequency factor F_j is [1,1] since all the sessions execute at the nominal frequency. That is to say, $F_j = 1$, since $f(S_j) = f_{nom}, \forall j \in \{1, 2, ..., k\}$.

The test length of a session S_j, owing to the introduction of F_m, can be expressed as: $L_{S_j} = \max\{L_{t_i}\}/\min\{F_m, (P_{max}/\sum P_{t_i})\}, \forall t_i \in S_j$. The total test time for k sessions is given by, $\sum_{j=1}^{k} [\max\{L_{t_i}\}/\min\{F_m, (P_{max}/\sum P_{t_i})\}], \forall t_i \in S_j$. From the expression for total time, we infer the following:

(a) As $F_m \to 0$, total test time $\to \infty$.

(b) As $F_m \to \infty$,

$$\text{total test time} \to \sum_{j=1}^{k} [\max\{L_{t_i}\}/(P_{max}/\sum P_{t_i})].$$

Different test schedules are obtained for different values of F_m, as shown in Table III.

TABLE III
OPTIMAL SCHEDULES OBTAINED FOR THE TCG IN FIGURE 2 FOR DIFFERENT VALUES OF F_m.

$F_m = 1$		$F_m = 3$		$F_m = 4$	
Sessions	Test Length	Sessions	Test Length	Sessions	Test Length
T1,T6	100	T1,T6	75	T1	50
T2,T5	10	T2,T5	7.5	T2	2.5
T3,T4	10	T3	3.33	T3	2.5
		T4	1.65	T4	1.25
				T5	5
				T6	25
	120		87.49		86.25

IV. CASE STUDY

In this section, we demonstrate our proposed method on an SoC, ASIC Z, as a case study. The ASIC Z was presented by Zorian [15]. It consists of RAM, ROM and Logic blocks along with a register file. The blocks, along with their test time and test power are shown in Figure 3. The estimation of optimal test time for this system was made by Chou *et al* [14] as 331 units and by Larsson and Peng [22] as 300 units (Table IV). Solving the ILP model M1 for the ASIC Z also yields the same result as that of Larsson and Peng [22].

By including the frequency factor and solving the ILP model *M2* for ASIC Z, we find that the lower bound for the optimal test time is 220.19 units (Table V).

In the previous section, we have seen that the expression for total test time, when realistic constraints such as power limit of individual tests are considered, is given by $\sum_{j=1}^{k} [\max\{L_{t_i}\}/\min\{F_m, (P_{max}/\sum P_{t_i})\}], \forall t_i \in S_j$. From the above expression for total test time, we observe that the total test time is influenced by the choice of P_{max} and F_m. The optimal test time as a function of F_m and P_{max} is shown in Figure 4.

From the plot in Figure 4(a), we observe that the optimal test time for F_j in the range [0,1], i.e., $0 \leq F_j \leq 1$, is 285.3. By reducing the frequency of operation of a session, although the test length increases, the power dissipated reduces and this allows us to schedule more number of tests. Hence, by decreasing the clock frequency we are increasing the concurrent execution of tests. For the ASIC Z, this strategy results in a total test time of 285.3, which is still better than the original optimal

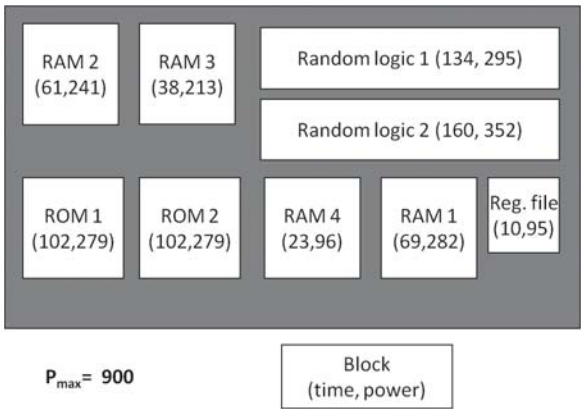


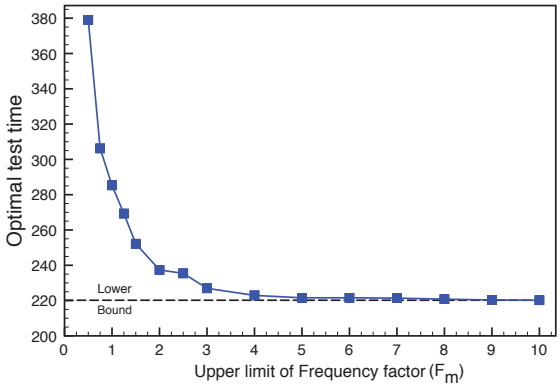
Fig. 3. The components of ASIC Z, and their test time (in arbitrary units) and test power (in mW).

TABLE IV
A COMPARISON OF OPTIMAL TEST TIMES OBTAINED, IN THE PAST, FOR ASIC Z.

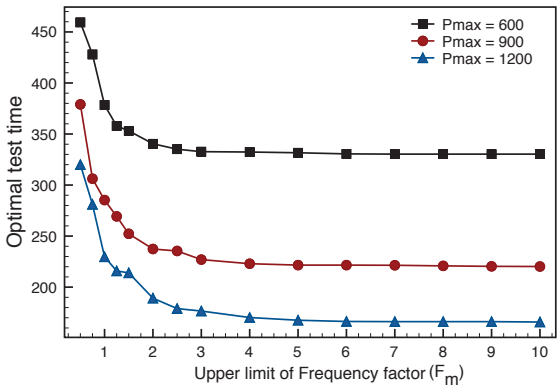
Chou *et al.* [14]		Larsson & Peng [22]	
Test Time	Blocks	Test Time	Blocks
69	RAM1,RAM3, RAM4,RF	160	RL2,RL1,RAM2
160	RL1,RL2	102	RAM1,ROM1, ROM2
102	ROM1,ROM2, RAM2	38	RAM3,RAM4,RF
331		300	

TABLE V
THE LOWER BOUND FOR TOTAL TEST TIME OF ASIC Z

Session	**Block**	**Freq. factor**	**Test Length**
1	RAM1	3.19	21.62
2	RAM2	3.734	16.33
3	RAM3	4.225	8.993
4	RAM4	9.374	2.453
5	ROM1	3.22	31.62
6	ROM2	3.22	31.62
7	RL1	3.05	43.92
8	RL2	2.556	62.58
9	RF	9.473	1.06
		Total test time =	220.19



(a)



(b)

Fig. 4. The plot of optimal test time v/s F_m (a) with the lower bound, for $P_{max} = 900$ (b) for three different power budgets

value of 300. The test schedule is shown in Table VI. The number of tests that can be executed concurrently, however, is limited by resource constraints, such as TAM width availability.

TABLE VI
TEST SCHEDULE OBTAINED FOR A FREQUENCY FACTOR LESS THAN 1

Session	Block	Freq. factor	Test Length
1	ROM1,2 & RL1,2	0.75	214.224
2	RAM1,2,3,4 & RF	0.97	71.07
		Total test time =	285.294

A. Imposing limits on the clock frequency of individual cores

So far we have assumed that test length and power are the only attributes for the test of a core and that the frequency of operation of a session/test is only power constrained, i.e., the frequency of a session is such that the power consumed during the session does not exceed P_{max}. However, for each core, there exists a maximum operational frequency that satisfies the structural constraints, such as critical path etc., and power constraints (maximum power dissipation limit) of the core. To account for this, we characterize the test t_i for each core using three parameters: test time (L_{t_i}), test power (P_{t_i}) and the core's maximum operational frequency (f_{t_i}). Let us assume that the clock frequency of the SoC, at which L_{t_i} and P_{t_i} are characterized, is decided by the maximum operational frequency of the slowest core in the SoC (say f_0). We also assume that $f_{nom} = f_0$, i.e., the nominal frequency of operation assumed, in Section II, for the classic case, is actually the frequency of the slowest core in the SoC. Hence, frequency factor of a session, $F_j = \frac{f(S_j)}{f_0}$. For the slowest core, $F_j = 1$.

Consequentially, a third constraint is added to the ILP model *M2*, which restricts the maximum value of the operational frequency of a session to that of the slowest test(core) scheduled to execute in that session, or $f(S_j) \leq \min\{f_{t_i} | \forall t_i \in S_j\}$. Let $f_{j_{max}}$ be the upper limit to which a session's frequency of operation, $f(S_j)$ can be increased. The maximum speed at which a session can be executed is decided by the speed of the slowest core. Hence, $f_{j_{max}} = \min\{f_{t_i} | \forall t_i \in S_j\}$. Let $F_{j_{max}} = \frac{f_{j_{max}}}{f_0}$. This implies $F_j \leq F_{j_{max}}, \forall j \in \{1, 2, .., k\}$. However, from the first constraint in ILP model *M2*, $F_j \leq P_{max}/P_{S_j}, \forall j \in \{1, 2, .., k\}$. Hence, $F_j \leq \min\{F_{j_{max}}, P_{max}/P_{S_j}\}, \forall j \in \{1, 2, .., k\}$.

Based on their test length and test power, we have assigned the maximum clock frequency, normalized with respect to the slowest, to the blocks present in ASIC Z, as shown in Table VII. Since the component Random Logic 2 (RL2) has the longest test length and consumes the most power, it is assumed to be the slowest.

TABLE VII
NORMALIZED MAXIMUM FREQUENCY VALUES FOR ASIC Z COMPONENTS

Block	Norm. Max. frequency
RAM1	1.75
RAM2	2
RAM3	3
RAM4	5
ROM1	1.5
ROM2	1.5
RL1	1.2
RL2	1
RF	8

The plot of total test time as the range of the frequency factor is varied, is shown in Figure 5, in comparison with the plot in Figure 4(a), where the frequency factor was limited only by the power constraint. For the frequency values assigned, the most optimal test time obtained was 268.3. The optimal test schedule achieved for this result is shown in Table VIII. The obtained results is, of course, specific to the assigned maximum frequency values. As a result, different optimal test times can be obtained for different assignments. However, the worst case scenario would be when the clock frequency for all the cores is same as that of the slowest (f_0). In other words, if $F_j = 1$ for all the sessions, then the optimal test time would be the same as the existing solution of 300.

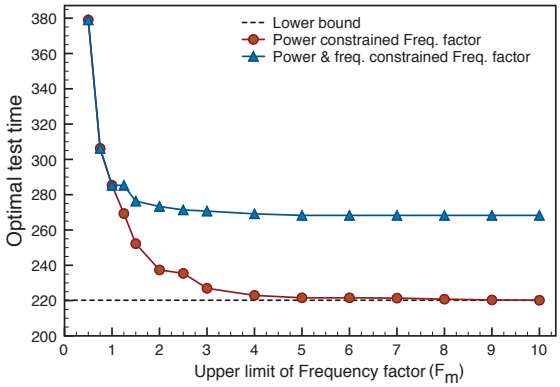


Fig. 5. The plot of optimal test time v/s F_m when F_j is constrained by both SoC's power budget and frequency limit of the core

TABLE VIII
TEST SCHEDULE ACHIEVED WHEN THE FREQUENCY FACTOR IS LIMITED BY BOTH POWER AND FREQUENCY CONSTRAINTS

Session	Block	Freq. factor	Test Length
1	RAM1,ROM2	1.5	68
2	RAM2,RAM3	1.98	30.77
3	RAM4,RF	4.71	4.88
4	ROM1,RL1,RL2	0.97	164.624
		Total test time =	268.274

V. CONCLUSION

We have proposed a method of minimizing the total test time of an SoC, for a given power budget, by incorporating variable test clock frequency for each test session. The existing Integer Linear-Programming model for the classic test scheduling problem is modified to incorporate a term, frequency factor, that corresponds to the operating frequency of each session. We have demonstrated this idea on ASIC Z, a system that has been frequently used by previously published literature as a benchmark. In the case study, we show that a considerable improvement of the total test time can be obtained by controlling the operational frequency of the test sessions. For the assumption that the test clock frequency is restricted only by the power budget of the SoC, the improvement achieved on existing solution is 27% and for the assumption that the test clock frequency is limited not only by the power budget of the SoC but also by maximum operational frequency limit of individual cores, the total test time obtained is 12% better than the existing solution. We have also proved that the lower bound on the test time can be achieved by scheduling one test per session, while increasing the session's frequency of operation such that the test power of each session reaches the power budget. Here, the assumption is that the increase in the frequency of operation is limited only by the power budget of the SoC and that the power of individual tests(cores) can be increased to equal the SoC's power budget. At-speed testing, wherein the tests are executed at the rated-clock speed, may seem to not gain much advantage from our proposed technique, since the test is expected to operate constantly at the maximum frequency. However, Moghaddam et al have shown, in [23] that reducing the number of transitions in the scan chains during scan load will allow the scan shift frequency to be increased, thus resulting in a lower test time. Future work includes applying the test time optimization method, proposed in this paper, to other existing benchmark SoCs and extending the idea for the case of partitioned test scheduling.

REFERENCES

[1] S. K. Goel and E. J. Marinissen, "Effective and efficient test architecture design for socs," in *International Test Conference*, 2002.

[2] K. Chakrabarty, "Optimal test architectures for system-on-a-chip," *ACM Transactions on Design Automation of Electronic System*, vol. 6, pp. 26–49, January 2001.

[3] ——, "Test scheduling for core-based systems using mixed-integer linear programming," *IEEE Transactions on Computer-aided design of integrated circuits and systems*, vol. 19, no. 10, pp. 1163–1174, 2000.

[4] W. Zou, S. M. Reddy, I. Pomeranz, and Y. Huang, "Soc test scheduling using simulated annealing," in *Proceedings of the 21st IEEE VLSI Test Symposium*, 2003.

[5] V. Iyengar, K. Chakrabarty, and E. J. Marinissen, "On using rectangle packing for soc wrapper/tam co-optimization," in *Proceedings of the 20 th IEEE VLSI Test Symposium*, 2002.

[6] J. Pouget, E. Larsson, Z. Peng, M. L. Flottes, and B. Rouzeyre, "An efficient approach to soc wrapper design, tam configuration and test scheduling," in *Proceedings of the Eighth IEEE European Test Workshop*, 2003.

[7] D. Zhao and S. Upadhyaya, "Power constrained test scheduling with dynamically varied tam," in *Proceedings of the 21st IEEE VLSI Test Symposium*, 2003.

[8] E. Larsson, *Introduction to Advanced System-on-Chip Test Design and Optimization*. Springer, 2005.

[9] Y. Huang, S. M. Reddy, W. T. Cheng, P. Reuter, N. Mukherjee, C. C. Tsai, O. Samman, and Y. Zaidan, "Optimal core wrapper width selection and soc test scheduling based on 3-d bin packing algorithm," in *Internation Test Conference*, 2002.

[10] P. Girard, N. Nicolici, and X. Wen, *Power-Aware Testgin and Test Strategies for Low Power Devices*. Springer, 2010.

[11] N. Nicolici and B. M. Al-Hashimi, *Power-Constrained Testing of VLSI Circuits*. Springer, 2003.

[12] V. Iyengar and K. Chakrabarty, "Precedence-based, preemptive and power constrained test scheduling for system-on-chip," in *Proceedings of VLSI Test Symposium*, 2001.

[13] E. Larsson and H. Fujiwara, "Power constrained preemptive tam scheduling," in *Proceedings of the Seventh IEEE European Test Workshop*, 2002.

[14] R. M. Chou, K. K. Saluja, and V. D. Agrawal, "Scheduling tests for vlsi systems under power constraints," *IEEE Trans. VLSI Systems*, vol. 5, no. 2, pp. 175–185, June 1997.

[15] Y. Zorian, "A distributed control scheme for complex vlsi devices," in *Proc. VLSI Test Symp.*, April 1993, pp. 4–9.

[16] K. Chakrabarty, "Test scheduling for core-based systems," in *Proceedings of ICCAD*, 1999.

[17] V. Iyengar, K. Chakrabarty, and E. J. Marinissen, "Test wrapper and test access mechanism co-optimization for system-on-chip," *J. Electronic Testing: Theory and Applications*, vol. 18, pp. 211–228, March 2002.

[18] P. Shanmugasundaram, "Test time optimization in scan circuits," Master's thesis, Auburn University, 2010.

[19] C. Bron and J. Kerbosch, "Finding all cliques of an undirected graph," *Communications of ACM*, pp. 575–577, 1973.

[20] *Python Implementation of Bors-Kerbosch*, https://github.com/alanmc/python-bors-kerbosch.

[21] *GLPK (GNU Linear Programming Kit)*, http://www.gnu.org/software/glpk/.

[22] E. Larsson and Z. Peng, "An integrated framework for the design and optimization of soc test solutions," *JETTA, Special Issue on Plug-and-Play Test Automation for System-On-chip*, vol. 18, pp. 385–400, 2002.

[23] E. K. Moghaddam, J. Rajski, S. M. Reddy, and J. Janicki, "Low test data volume low power at-speed delay tests using clock-gating," in *Proc. of Asian Test Symp.*, Nov 2011, pp. 267–272.