Power-Time Tradeoff in Test Scheduling for SoCs

Mehrdad Nourani and **James Chin**

Center for Integrated Circuits & Systems
The University of Texas at Dallas
Richardson, TX 75083-0688
{nourani,jchin}@utdallas.edu

ABSTRACT

We present a test scheduling methodology for core-based system-on-chips that allows tradeoff between system power dissipation and overall test time. The basic strategy is to use the power profile of non-embedded cores to find the best mix of their test pattern subsets that satisfy the power and/or time constraints. An MILP formulation is presented to globally perform the power-time tradeoff and produce the SoC test schedule. Many constraints including peak/average power of cores, time/sequencing requirements, and ATE pin limitation are also incorporated within this formulation.

I. INTRODUCTION

A. Motivation

Power consumption during test has become a challenge for test engineers. Power is a heavily data-dependent factor that is often checked only for normal-mode inputs (normal patterns). Empirically, the test-mode inputs (test patterns) could be substantially irregular and stimulate the internal circuitry to consume excessive power never experienced in the normal mode. Creating such hot spots can burn the chip during test or lower its reliability and lifetime [1]. This difficulty motivated us to work on a test scheme that allows direct power-time tradeoff during test. Assuming the test patterns and their corresponding power dissipation profiles are known, the formulation optimizes the SoC's power under time constraints or optimizes test time under cores' power constraints.

B. Prior Work

Using pre-designed cores to implement a system-on-chip can significantly shorten the design turnaround time and time-to-market. However, this advantage brings with itself many challenges for the SoC testing. These challenges include: accessing cores buried deep inside, with limited IO pins (design of test access mechanism) [2] [3], test scheduling to shorten the overall test time and manage the activities efficiently [4], testing the interconnects among the cores [5] [6] and power dissipation of cores and SoC during test. More specifically, due to popularity of low-power SoC applications special attention was paid in the literature to address concerns about power dissipation during test. The work in [7] presents a heuristic, based on the compatibility graph, to limit the power dissipation during test. In [8], an algorithm for power-constrained scheduling problem is presented. A method for determining preemptive and power-constrained schedules is discussed in [9].

C. Contribution and Paper Organization

The main contribution of our work is a test scheduling methodology for core-based SoCs that allows the user to perform power-time tradeoff. This is specifically advantageous for large multi-core SoCs that require large set of deterministic or pseudorandom test patterns. A mixed integer linear programming (MILP) formulation takes information about power consumption of non-embedded cores (i.e. power profile) into account and optimizes the test schedule to achieve minimum overall test time or minimum power consumption. This method considers the peak and/or average power consumption of cores, time/sequencing requirements and the ATE (automatic test equipment) pin limitation as constraints.

The rest of this paper is organized as follows. We review the basic factors contributing to power consumption and concept of core-based SoC design in Section II. The simulation-based method to obtain the power profile of cores and their role in power-time tradeoff are discussed in Section III. The basic MILP formulation, to minimize both the test time and the power consumption, is explained in Section IV. The experimental results are discussed in Section V. Finally, the concluding remarks are in Section VI.

II. BACKGROUND

A. Importance of Power Analysis

The great demand for wireless and mobile communication systems, and the increase of frequency and density of VLSI circuits made the power dissipation as important as area and delay, in design, synthesis and testing of such applications. In CMOS technology power dissipation is mostly due to capacitive charging and discharging of the nodes [10].

Power dissipation is a data-dependent characteristic. Researchers argued that different estimation algorithms that measure the switching probability and then estimate dynamic power, can be evaluated by their ways of addressing these two factors: 1) multiple-bit input switching (transition); and 2) the correlation among signals. Ignoring each of these two results in lack of accuracy. There are analytical methods, such as [11] [12] simulation-based, such as [13] [14], to estimate the switching probability. A survey of these methods is presented in [15].

B. SoC Design & Test Paradigm

In order to cope with the growing complexity of modern systems, designers often use pre-designed, reusable megacells known as cores. A core is a highly complex logic block which is fully defined in terms of its behavior, and is also predictable

and reusable. Typical cores include memories, microprocessors, datapath-controller pairs, etc. Core-based SoC design strategies help companies significantly reduce the time-to-market and design cost for their new products. The Semiconductor Industry Association's National Technology Roadmap [16] predicts the percentage of reused cores in SoCs rising from 40% in 1997 to 80% in 2006 while it expects a 50% reduction in time-to-market.

Unfortunately, there are difficulties in testing core-based SoCs. Some difficulties stem from the sheer complexity inherent in putting an entire system's functionality on a single chip, which makes it harder to control and observe what is happening in the circuits embedded deeply inside the chip [17][2][3]. Other difficulties arise when the complete information about the cores is not available to the system designer. This happens because cores are often purchased from the third party vendors anxious to protect their intellectual property.

III. POWER PROFILING FOR NON-EMBEDDED CORES

A. Concept

To be able to control the SoC's power consumption during test first we need to have a clear picture of cores' power consumptions as their corresponding test patterns are applied. Specifically, at the logic gate level, a circuit simulator computes the functional values of all nodes and therefore the switching activities become available. Practically, many power analysis tools, such as simulators in Synopsys [18] do this by adding some analytical computation to the original gate level simulation.

Tracing the power consumption for each transition from one pattern to the next is not practical. Therefore, we propose to approximate power analysis by putting some breakpoints in applying the patterns. These breakpoints effectively partition a test pattern set to some subsets and the formulation can look for the best timing to apply each. Inserting $m - 1$ breakpoints partitions the set of n patterns to m subsets, e.g. $S_1, \cdots, S_k, \cdots, S_m$, such that $\sum_{k=1}^{m} |S_k| = n$. Without loss of generality, we assume that these breakpoints are drawn in such a way to create subsets with equal number of patterns, i.e. $|S_k| = |S| = cte \quad \forall k$.

Figure 1 shows the power consumption of a small multiplier core for $n = 12$ test patterns. Two power profiles corresponding to subset size ($|S|$) of 4 and 2 are shown. Obviously, smaller subset size means more accuracy and also more computational time during optimization. For this example, using first and second profile, we need to integrate $n/|S|$ numbers in the formulation, i.e. $\{3, 8, 7\} \, mWatt$ and $\{4.1, 4.3, 7.8, 8.3, 7.5, 7.1\} \, mWatt$ for profile 1 and 2, respectively.

When multiple cores exist in the system, still one number, i.e. the greatest common divisor, will be chosen for $|S|$. Parameter S helps our formulation to perform scheduling and blend different partitions for power-time tradeoffs. Note that in our discussion each *time step* is equivalent to the number of cycles, in test mode, required to send $|S|$ patterns. More specifically, the basic test scheduling time step (unit) will be equivalent to time needed to apply $|S|$ test patterns. If the speeds of cores during test are different, we still use one parameter $|S|$ but it corresponds to different number of test patterns for different cores. For example, assume the length of scan chains in two cores are the same and accessing these chains have no other restriction. $|S|$ patterns
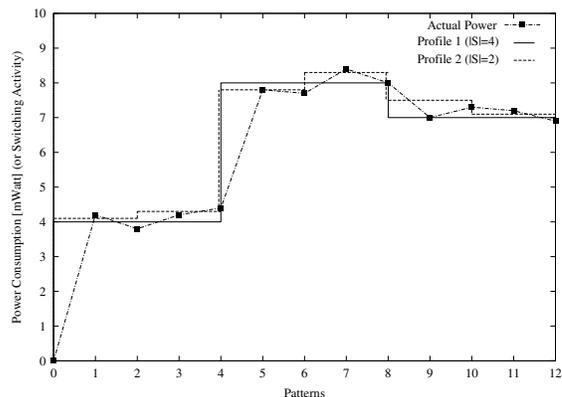

Fig. 1. The concept of profiling for a small example.

of a core with frequency f_1 is time-wise equivalent to $\frac{f_2}{f_1} \cdot |S|$ patterns of another core with frequency f_2.

There are analytical optimization techniques as to how to increase accuracy of such profiling. This is beyond the scope of this paper but interested reader can refer to approximation and regression techniques [19] [20] [21].

B. Power-Time Tradeoff

Having the power profiles of cores, we are ready now to devise an optimization technique to do tradeoff between power (peak or average) and test time. This optimization technique should also produce a test schedule to control power during test.

As pointed out earlier, dynamic power dissipation forms the significant part of overall power consumption of digital SoCs. Dynamic power is a data-dependent factor that depends on switching activities which itself depends on number of bit transitions between the patterns. Therefore to be able to use the power profile, we should preserve the order of pattern for at least some partitions (e.g. for subsets S_k) of patterns.

Just to show how power-time tradeoff offers different test schedule we use a small SoC made of two cores. Figure 2 shows the power profiles of two cores with subset size of $|S| = 4$. Core 1 and 2 require 8 and 12 test patterns, respectively. Figure 3 shows four test schedules for the SoC. In this figure the shaded and white bars correspond to the power profiles of Core 1 and Core 2 (reflected also in Figure 2), respectively. For simplicity, we assumed there is no limitation on available ATE pins. Test schedules shown in Figure 3(a) and (b) correspond to minimum test time subject to the SoC's peak power of 13 and 10, respectively. The test schedule in Figure 3(c) is generated with constraint on the overall average overall power, that is $\frac{\sum_k P_{j,k}}{(e_j - b_j) + 1} = \frac{7 + 5 + 4 + 8 + 7}{(5-1) + 1} = 6.20$. Note that for *Core_j*, the average power of testing a core depends on power dissipation caused by each set ($P_{j,k}$) and its start/end time step (b_j / e_j) and not directly on the overall test time. For example here, the average power for core 1 and core 2 is $\frac{7+5}{(2-1)+1} = 6.00$ and $\frac{4+8+7}{(5-3)+1} = 6.33$, respectively. Finally, the schedule in Figure 3(d) takes into account the average power of individual cores. In this schedule, the average power of core 1 and core 2 will be $\frac{7+5}{(3-1)+1} = 4.00$ and $\frac{4+8+7}{(5-2)+1} = 4.75$, respectively that satisfy the constraints.
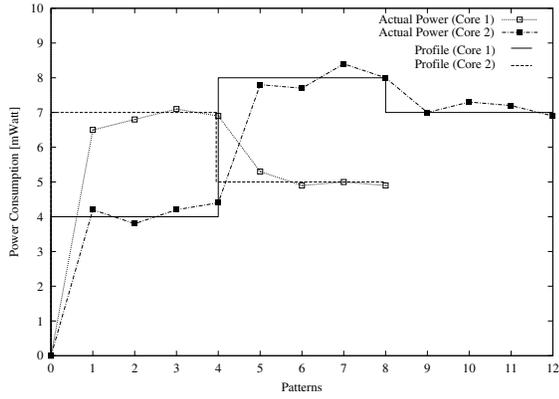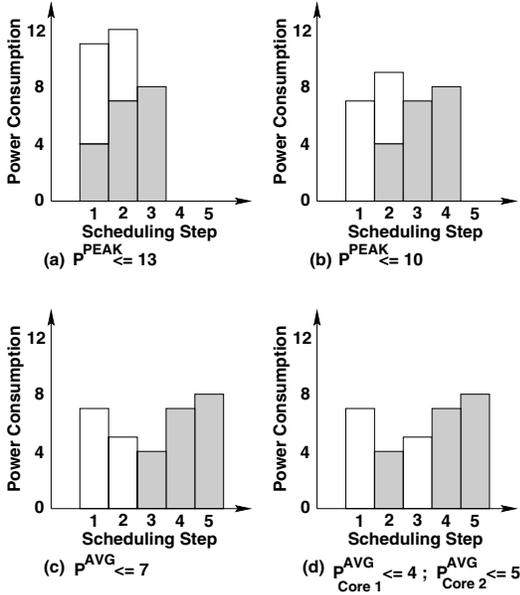
Fig. 2. A SoC example: power profile of two cores.

Fig. 3. A SoC example: four test schedules.

IV. THE MILP FORMULATION

In this section we discuss the integer linear programming formulation whose objective function can reflect overall scheduling time, total power limit or a weighted mix of both.

A. Constants

There are overall ten constants to be used in the formulation:

$$
\left\{
\begin{array}{ll}
N^{CORE} & \text{Total number of cores} \\
N_j^{SET} & \text{Total number of test pattern subsets for } Core\_j \\
P_{j,k} & \text{Peak power dissipation when subset } S_k \text{ tests } Core\_j \\
P^{PEAK} & \text{Maximum peak power of SoC} \\
P^{AVG} & \text{Maximum average power of SoC} \\
P_j^{AVG} & \text{Maximum average power for } Core\_j \\
N^{PIN} & \text{Total number of pins available to test SoC} \\
N_{j,k}^{PIN} & \text{Total number of pins required to test } Core\_j \\
& \text{when subset } S_k \text{ is applied} \\
W_t & \text{Optimization weight for total test scheduling time} \\
W_p & \text{Optimization weight for total power consumption}
\end{array}
\right.
$$

For simplicity in expressing the constraints, we also define

T^{MAX} as the upper bound for the test set time steps requires for test schedule. This is a scenario in which cores are tested in sequence:

$$
T^{MAX} = \sum_{j=1}^{N^{CORE}} N_j^{SET}
$$

Obviously, T^{MAX} is not independent of other constants. We use it only for further clarification of constraints. Note that, the above list of constraints shows that we take into account the pin limitations of ATE and the pin requirement for each core. This makes the formulation more practical. Of course if such information is not available or bounding, the constants can be replaced with a large number.

B. Variables

■ There are two global variables, $total\_time$ and $total\_power$ denoting overall test scheduling time and total power consumption of the system at any time step, respectively. $total\_power$ can take a real value since all of the power constants (P^{PEAK}, P^{AVG}, $P_{j,k}$, etc.) can be real numbers. This is why our formulation is a mixed integer linear programming (MILP). We need an additional set of variables as defined below:

$$
t_{j,k,l} = \left\{
\begin{array}{ll}
1 & \text{if } Core\_j \text{ is scheduled to receive its test data} \\
& \text{subset } S_k \text{ at time step } l \\
0 & \text{otherwise}
\end{array}
\right.
$$

For the purpose of simplicity in writing the equations and constraints, we define $c_{j,l}$ variables:

$$
c_{j,l} = \sum_{k=1}^{N_j^{SET}} t_{j,k,l}
$$

$c_{j,l} = 1$ means that $Core\_j$ is tested at time l. Note that $c_{j,l}$ is not independent of other variables. We use it only for further clarification in writing the constraints.

■ To be able to compute average power for each core, we need to know when its testing begins and when it ends. Let b_j and e_j denote time step index that Core_j receives the first and last test pattern sets, respectively.

C. Objective Function

■ The objective is to minimize a linear function of weighted test time and power consumption:

$$
(W_t \cdot total\_time) + (W_p \cdot total\_power)
$$

Selecting normalized weights for W_t and W_p (e.g. $W_t = \frac{r_t}{r_t+r_p}$ and $W_p = \frac{r_p}{r_t+r_p}$ for arbitrary positive values of r_t and r_p) imply the importance of these factors in the optimization process from the user point of view. For example, (W_t, W_p) values of $(1,0)$ and $(0,1)$ correspond to time and power optimization, respectively. $(W_t, W_p) = (0.5, 0.5)$ shows that both factors are seen equally important. By changing these two weights we can explore different options and see the tradeoff between time and power to select the best case that fits our constraints and need.

D. Constraints

Based on the constants defined earlier, we use three indices (j, k and l) to define the variable. Their ranges are: $1 \leq j \leq N^{CORE}$, $1 \leq k \leq N_j^{SET}$ for *Core_j* and $1 \leq l \leq T^{MAX}$. The required variables will be as follows.

- Variables $t_{j,k,l}$ should be 0 or 1:

$$t_{j,k,l} \leq 1 \quad \forall j, k, l \tag{1}$$

- Variables $c_{j,l}$ indicate that *Core_j* is tested at time step l:

$$c_{j,l} \leq 1 \quad \forall j, l \tag{2}$$

- Each scheduling time step used should be within the given bound:

$$total\_time \leq T^{MAX} \tag{3}$$

- Define $c_{j,l}$ based on $t_{j,k,l}$ for simplicity:

$$c_{j,l} = \sum_{k=1}^{N_j^{SET}} t_{j,k,l} \quad \forall j, l \tag{4}$$

- The assigned time step (e.g. $l * c_{j,l}$) to test each core should stay within the overall scheduling time:

$$l * c_{j,l} \leq total\_time \quad \forall j, l \tag{5}$$

- All test pattern subsets for each core must be applied during the schedule:

$$\sum_{l=1}^{T^{MAX}} c_{j,l} = N_j^{SET} \quad \forall j \tag{6}$$

- Total power should stay within the limit at any given test time step:

$$total\_power \leq P^{PEAK} \tag{7}$$

- Total power of the selected cores to be tested at each time step should be within the limit:

$$\sum_{j=1}^{N^{CORE}} \sum_{k=1}^{N_j^{SET}} P_{j,k} * t_{j,k,l} \leq total\_power \quad \forall l \tag{8}$$

- Any test pattern subset must be scheduled only once:

$$\sum_{l=1}^{T^{MAX}} t_{j,k,l} = 1 \quad \forall j, k \tag{9}$$

- If there are information and limit on the ATE pins, they can be also incorporated within the constraints as follows. The total number of ATE pins used at the same time is limited:

$$\sum_{j=1}^{N^{CORE}} \sum_{k=1}^{N_j^{SET}} N_{j,k}^{PIN} * t_{j,k,l} \leq N^{PIN} \quad \forall l \tag{10}$$

- Each core has its own average power limit:

$$\sum_{k=1}^{N_j^{SET}} P_{j,k} \leq P_j^{AVG} * (e_j - b_j + 1) \quad \forall j \tag{11}$$

- Total average power is also limited:

$$\sum_{j=1}^{N^{CORE}} \sum_{k=1}^{N_j^{SET}} P_{j,k} \leq P^{AVG} * total\_time \tag{12}$$

- Variables e_j and b_j are bounded:

$$l * c_{j,l} \leq e_j \leq total\_time \quad \forall j, l \tag{13}$$

$$1 \leq b_j \leq (T^{MAX} + 1) - (T^{MAX} - l + 1) * c_{j,l} \quad \forall j, l \tag{14}$$

We need also to make sure that variables e_j/b_j get the maximum/minimum of scheduled time steps. To do this we add this term also in the objective function that, independent of total_time and total_power, maximize/minimize these variables:

$$\sum_{j=1}^{N^{CORE}} (e_j - b_j)$$

- For some cores (e.g. combinational circuits) the order of test pattern subsets can be changed. This may change the power profile accuracy slightly due to transition changes on patterns in border of subsets. But for large S and large number of subsets, this is negligible. On the other hand, for others (e.g. sequential circuits or when we need higher power accuracy) we may prefer to preserve the order of test pattern subsets. In such cases, the subset S_k must be applied before subset S_{k+1}:

$$\sum_{l=1}^{T^{MAX}} l * t_{j,k,l} < \sum_{l=1}^{T^{MAX}} l * t_{j,k+1,l} \quad \forall j, k \tag{15}$$

E. The Complete MILP Formulation

The complete MILP formulation is obtained by combining all constraints expressed by Equations 1 through 15:

Minimize $\quad (W_t \cdot total\_time) + (W_p \cdot total\_power) + \sum_{j=1}^{N^{CORE}} (e_j - b_j)$
subject to:

(1) $\quad t_{j,k,l} \leq 1 \qquad \forall j, k, l$

(2) $\quad c_{j,l} \leq 1 \qquad \forall j, l$

(3) $\quad total\_time \leq T^{MAX}$

(4) $\quad c_{j,l} = \sum_{k=1}^{N_j^{SET}} t_{j,k,l} \qquad \forall j, l$

(5) $\quad l * c_{j,l} \leq total\_time \qquad \forall j, l$

(6) $\quad \sum_{l=1}^{T^{MAX}} c_{j,l} = N_j^{SET} \qquad \forall j$

(7) $\quad total\_power \leq P^{PEAK}$

(8) $\quad \sum_{j=1}^{N^{CORE}} \sum_{k=1}^{N_j^{SET}} P_{j,k} * t_{j,k,l} \leq total\_power \qquad \forall l$

(9) $\quad \sum_{l=1}^{T^{MAX}} t_{j,k,l} = 1 \qquad \forall j, k$

(10) $\quad \sum_{j=1}^{N^{CORE}} \sum_{k=1}^{N_j^{SET}} N_{j,k}^{PIN} * t_{j,k,l} \leq N^{PIN} \qquad \forall l$

(11) $\quad \sum_{k=1}^{N_j^{SET}} P_{j,k} \leq P_j^{AVG} * (e_j - b_j + 1) \qquad \forall j$

(12) $\quad \sum_{j=1}^{N^{CORE}} \sum_{k=1}^{N_j^{SET}} P_{j,k} \leq P^{AVG} * total\_time$

(13) $\quad l * c_{j,l} \leq e_j \leq total\_time \qquad \forall j, l$

(14) $\quad 1 \leq b_j \leq (T^{MAX} + 1) - (T^{MAX} - l + 1) * c_{j,l} \qquad \forall j, l$

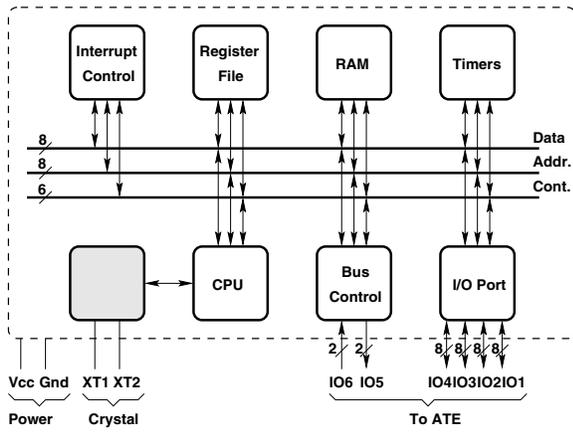(15) $\quad \sum_{l=1}^{T^{MAX}} l * t_{j,k,l} < \sum_{l=1}^{T^{MAX}} l * t_{j,k+1,l} \qquad \forall j, k$

Fig. 4. Intel 8051 microcontroller block diagram.

Note that the order of N^{CORE} and T^{MAX}, which determines the number of constraints, makes this formulation of a manageable size such that almost any MILP software package can solve it. Specifically, if T^{MAX} is large we can tradeoff accuracy with faster MILP search time by considering larger size of test pattern subsets.

Our formulation is comprehensive and flexible. Briefly speaking, constraints (1) through (9) form a complete (mandatory for convergence) set for peak power control. Constraints (11), (13) and (14) limit the average power consumption of each core and can be removed if there is no such limit. Three constraints can be independently added based on user's choice. They are constraint (10) on the ATE pin limit, constraint (12) on the average power limit of the concurrent running cores, and constraint (15) on the order of patterns within the pattern set for sequential circuits.

V. EXPERIMENTAL RESULTS

We used the ILOG *CPLEX* package from ILOG S. A., Inc. [22] to solve the MILP formulation. We wrote a C program to translate the ATE/SoC/Core constraints, user-defined specifications and power profile information to the MILP input format required by CPLEX. This C program generates the objective and constraints in *lp* format in a few seconds. This is a textual format recognized by many LP/ILP/MILP packages including *CPLEX* [22] and *lp_solve* [23].

To challenge a large example, we selected the Intel 8051 microcontroller. Figure 4 shows the block diagram of this microcontroller. Note that this figure shows the general topology of the system and does not necessarily reflect the exact bit widths of each core's input and output ports for the purpose of testing. Excluding the oscillator circuitry and RAM in test mode, the six cores in 8051 microcontroller have overall 20 input ports (with total bit width of 114 bits) and 13 output ports (with total bit width of 82 bits). The conventional 8051 has only six I/O terminals (overall bit width of 36) that can be be used to send test patterns to the cores. However, to be able to show the power-time tradeoff in our MILP formulation we assume no bottleneck exists on the core access mechanism and we can access multiple cores at the same time.

The cores of 8051 system are described in VHDL. Then, synthesis, test patterns generation power simulation and profiling are performed using Synopsys design toolset [18] using its generic library. Then, the MILP formulation has been applied to do the power-time tradeoff. The power-time tradeoff results are tabulated in Tables I, II and III for three different (W_t, W_p) weights.

The results of MILP formulation are tabulated using {total_time,**total_power**} notation. So, the normal face and boldface numbers reflect total scheduling time step and total power in $mWatt$, respectively. As shown in Table I, for given values of peak power and available pins, the system blends the test pattern subsets to satisfy average power constraint of each core as well as overall SoC. In Table II we changed the limit on overall peak power and the formulation increase or decrease the overall time steps to satisfy the constraints. Finally, as Table III reflects, having more ATE pins available for testing, shortens the test schedule since the test activities run in parallel as much as allowed by the peak and average power metrics. The average wall clock time are also reported using SUN ULTRA 10 workstation.

Three choices of (W_t, W_p) have been considered. For the first choice $(1, 0)$, the MILP formulation finds the fastest test schedule with no consideration of power. For $(0, 1)$, the formulation finds the schedule with the least power consumption with no effort to minimize time. Finally, choice of $(0.5, 0.5)$ implies that both factors are equally important. When both weights W_t and W_p take non-zero values, in general, the overall time and power values are often between the corresponding values obtained by $(1, 0)$ and $(0, 1)$ extremes. Note that the reason of having **122** as overall peak power of the system is the fact that according to power profile of *Core_3*, we have $P_{3,1} = 122$ and therefore, this is the lower bound that the formulation achieves. The CPLEX running time is longer approximately by a factor 2 to 10 when both weights take non-zero values.

To show how the test schedule will be affected when power constraints are changed, we have lowered the average power limit on cores by 75% and 50% and then ran the MILP. The
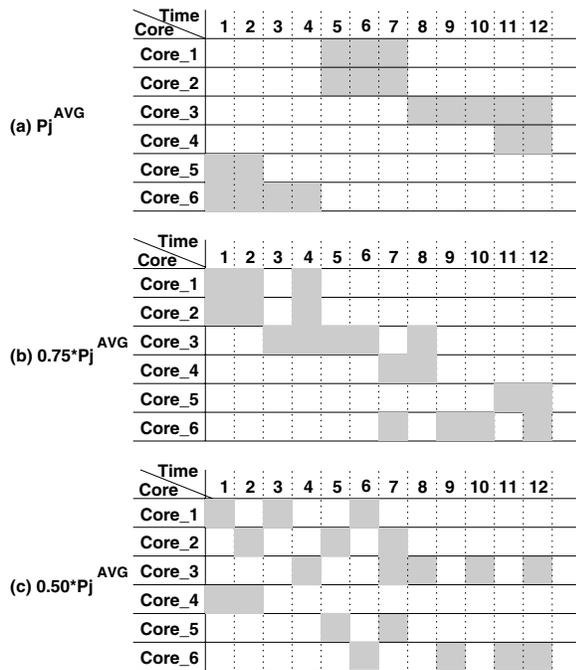
Fig. 5. Test schedules under different average power constraints for cores.

test schedule are shown in Figure 5. The shaded time steps are those that the core is receiving the test pattern subsets and thus is under test. Researchers used different techniques to control power. For example, [9] uses registers for intermediate storage to halt the test for cores. Our control mechanism is based on stopping the clock for cores to amortize power consumption to the desired limit. For example, overall testing time for *Core_1* for $P_1^{AVG} = 45$, $P_1^{AVG} = 0.75 * 45 = 33.75$ and $P_1^{AVG} = 0.50 * 45 = 22.25$ are 3, 4 and 6 time steps corresponding to ranges $[5, 7]$, $[1, 4]$ and $[1, 6]$, respectively. As Figure 5 demonstrates, the overall test time for each core and idle times inserted will change to avoid creating hot spots based on the given average power limit on the core.

The important practical point about our experimentation is: 1) the size of MILP formulation is quite reasonable such that almost any MILP package can solve it. The size is more sensitive to the number of cores and pattern subsets and thus the running time is approximately in order of $O\left(\left(N^{CORE} + T^{MAX}\right) * T^{MAX}\right)$ and 2) all three power factors (P^{PEAK}, P^{AVG} and especially $P_{j,k}$) significantly affect the results. The formulation provides a mechanism to globally balance power versus time.

To distinguish our work from similar approaches in the literature, we like to point out that the work in [7], [8], and [9] do not use detailed power profile of non-embedded cores to precisely control the transient peak power like ours. In our work, we use weights associated with test time and peak power in the ILP formulation. Consequently, both problems: 1) minimizing the test time under power constraints, and 2) minimizing the peak power under time constraints can be addressed by adjusting the weights. Additionally, our methodology is more flexible than others reported in the literature in handling various cores. More specifically, we can address cores that require fixed or variable order pattern sets and take the individual average power of cores into account. Finally, the maximum power dissipation of the entire SoC and the limit of the ATE pins are directly incorporated

VI. CONCLUSION

To be able to effectively perform power-time tradeoff during SoC testing we need to use an optimization technique that can blend various practical factors present in the test environment. We offered an MILP formulation that takes the power profile of non-embedded cores as inputs and generates the test schedule. This formulation is quite flexible as it takes into account various user-defined constraints on peak/average power, time bounds and ATE pin limitation.

ACKNOWLEDGEMENTS

REFERENCES

[1] P. Chen, L. Wu, G. Zhang and Z. Liu, "A Unified Compact Scalable ΔI_d Model for Hot Carrier Reliability Simulation," in Proc. *IEEE 37th Annual Intern. Reliability Physics Symp.*, pp. 243-248, 1999.
[2] E. Marinissen, R. Arendsen, G. Bos, H. Dingemanse, M. Lousberg and C. Wouters, "A Structured and Scalable Mechanism for Test Access to Embedded Reusable Cores," in *Proc. Intern. Test Conf. (ITC-98)*, pp. 284-293, 1998.
[3] M. Nourani and C. Papachristou, "An ILP Formulation to Optimize Test Access Mechanism in SoC Testing," in *Proceedings of the International Test Conference (ITC)*, pp. 902-910, Oct. 2000.
[4] K. Chakrabarty, "Test Scheduling for Core-Based Systems Using Mixed Integer Linear Programming," in *IEEE Trans. on CAD*, vol. 19 pp. 1163-1174, Oct. 2000.
[5] X. Bai, S. Dey and J. Rajski, "Self-Test Methodology for At-Speed Test of Crosstalk in Chip Interconnects," in Proc. *Design Automation Conf. (DAC'00)*, 2000, pp. 619-624.
[6] M. Nourani and A. Attarha, "Built-In Self-Test for Signal Integrity," in *Proceedings of the 38th Design Automation Conference (DAC)*, (Las Vegas, Nevada), pp. 792-797, June 2001.
[7] R. Chou, K. Saluja and and V. Agrawal, "Scheduling Tests for VLSI Systems Under Power Constraints," *Trans. on VLSI*, vol. 5, no. 2, pp. 175-185, June 1997.
[8] V. Muresan, X. Wang, V. Muresan, and M. Vladutiu, "List Scheduling and Tree Growing Technique in Power-Constrained Block Test Scheduling," in *Digest of Papers European Test Workshop*, pp. 27-32, 2000.
[9] V. Iyengar and K. Chakrabarty, "Precedence-Based, Preemptive, and Power-Constrained Test Scheduling for System-on-a-Chip," in Proc. of *VLSI Test Symposium (VTS)*, pp. 368-374, 2001.
[10] J. Rabaey, *Digital Integrated Circuits*, Prentice Hall, 1996.
[11] R. Marculescu, D. Marculescu, and M. Pedram, "Efficient Power Estimation for Highly Correlated Input Streams," *in Proc. of DAC*, pp. 628-634, 1995.
[12] P. Schneider, U. Schlichtmann, B. Wurth, "Fast Power Estimation of Large Circuits," *IEEE Design and Test of Computers*, pp. 70-78, 1996.
[13] J. Dunoyer, N. Abdallah, P. Bazargan, "A Symbolic Simulation Approach in Resolving Signals' Correlation," *Proc. of Simulation Symposium*, pp. 203-211, 1996 .
[14] J. Monterio, S. Devadas, A. Ghosh, K. Keutzer and J. White, "Estimation of Average Switching Activity in Combinational Logic Circuits Using Symbolic Simulation," *IEEE Trans. CAD* vol. 16, no. 1, pp. 121-127, 1997.
[15] F. Najm, "A Survey of Power Estimation Techniques in VLSI Circuits," *IEEE Transactions on VLSI*, vol. 2 Issue: 4, pp. 446-455, Dec. 1994.
[16] Semiconductor Industry Association, *The International Technology Roadmap for Semiconductors*, Sematech, Inc. 1999.
[17] N. Touba, B. Pouya, "Testing Embedded Cores Using Partial Isolation Rings," in *Proc. VLSI Test Symposium (VTS-97)*, pp. 1016, May 1997.
[18] Synopsys Design Analyzer, "User Manuals for SYNOPSYS Toolset Version 2000.05-1," Synopsys, Inc., 2000.
[19] R. Miller *Optimization : Foundations and Applications*, Wiley, 2000.
[20] J. Nocedal and S. Wright *Numerical Optimization*, Springer, 1999.
[21] T. H. Coerman, C. E. Leiserson and R. L. Rivest, *Introduction to Algorithms*, McGraw-Hill Company, 1990.
[22] ILOG S. A., Inc., "The User's Manual for ILOG CPLEX 6.5," 1999.
[23] M. Berkelaar, *lp_solve, version 3.0*, Eindhoven University of Technology (The Netherlands), Email: Michel@es.ele.tue.nl, 2000.