Peak Power Estimation: A Case Study on CPU Cores

Peak Power Estimation: a Case Study on CPU Cores

P. Bernardi, M. De Carvalho, E. Sanchez, M. Sonza Reorda
Dip. di Automatica e Informatica
Politecnico di Torino
Torino, Italy

A. Bosio, L. Dilillo, P. Girard, M. Valka
LIRMM
University of Montpellier 2 / CNRS
Montpellier, France

***Abstract*—High peak power consumption during test may lead to yield loss. On the other hand, reducing too much test power may lead to test escape. In order to overcome this problem, test power has to mimic the power consumed during functional mode, being as high as possible but not crossing the frontier of over-consumption. Measuring power consumption is a very time consuming activity; therefore many works in the literature focused on the indirect ways to provide power consumption estimation in a fast manner. In this paper we concentrate on a similar issue, concentrating our effort on devising a fast method for the identification and estimation of the peak power produced by test patterns. In particular we provide a detailed discussion on case studies related to peak power estimation of CPU cores when executing functional patterns; the proposed method uses the gate-level description of the CPU to identify a subset of time points over the entire test pattern that are showing the most significant peak power values. The proposed methodology has been validated on two case studies synthesized in a 65nm industrial technology.**

Keywords- Peak power estimation; Power-aware testing; At-speed delay fault testing; Functional power component

I. INTRODUCTION

Nowadays, electronic products design and manufacturing raise various issues that become increasingly more important with CMOS technology scaling. High operation speed and high frequency are mandatory requests. On the other hand, power consumption is one of the most significant constraints, especially due to large diffusion of portable devices. These needs influence not only the design of devices, but also the choice of appropriate test schemes that have to deal with production yield, test quality and test cost.

Testing for performance, required to catch timing or delay faults, is therefore mandatory, and it is often implemented through at-speed testing [1]. Although at-speed testing is mandatory for high-quality delay fault testing, its applicability is severely challenged by test-induced yield loss (i.e., overtesting), which may occur when a good chip is declared as faulty during at-speed testing, or when several successive power peaks occur during test and lead to chip damage. Several works, such as [2], analyzed these phenomena and demonstrated that they are related to test power consumption, which may be excessively high compared to power consumption during functional mode (i.e. functional power). Despite the fact that reduction of test power is mandatory to minimize the risk of yield loss, some experimental results have also proven that too much test power reduction may lead to test escape because of the under-stress of the circuit during test [1].

Such issues imply that testing must be aware of power consumption in order to achieve high test quality without affecting production yield. Therefore, the solution relies on mapping test power to the power consumed during functional mode. For this purpose, the knowledge of functional power for a given circuit under test (CUT) is required and has to be used as a reference for defining the power consumption (upper and lower) limits during power-aware delay test pattern generation.

In our previous work [3], we presented a flow to generate functional stimuli based on an evolutionary algorithm driven to maximize overall power consumption for a given circuit. The main drawback of that work is the evaluation of the power consumption for a given set of stimuli. Basically, we exploited a spice-like simulator [4] to evaluate the power consumption. A single evaluation at transistor-level may take weeks to provide an accurate result. Moreover, the proposed generation flow requires the evaluation of many stimuli in order to produce an effective one. Therefore, it is mandatory to estimate power consumption in a more efficient way.

In the literature [1], power consumption has always been associated to the toggling rate (i.e., the switching activity) of a circuit during application of a given set of stimuli; this kind of measurement is frequently used as power consumption indicator since it relies on the axiom that higher switching activity leads to higher power consumption.

In this paper, we first discuss about the accuracy of the switching activity measure to estimate the power consumption, in particular towards the identification of those circuit configurations that lead to the highest consumption. This analysis shows that the switching activity trend is coarsely matching the power trend, but a one-to-one relationship between switching activity and peak power is not usually observed. In other words, it is often observed that the time where the maximum peak power appears does not always correspond to the time point where maximum switching activity is observed.

Based on this consideration, we then propose a smarter power consumption estimation approach whose purpose is to identify a subset of time points along a pattern where it is likely to encounter significant power peaks. The method is based on a logic level switching activity- based analysis. This approach is useful to provide fast and effective power estimation that could be exploited for example to drive the automatic generation of

power hungry functional patterns for design validation purpose [5]. We validated the proposed methodology using two case studies: the Intel 8051 and the OpenRisc 1200, synthetized using a 65nm industrial technology. In both cases, the evaluation time for each input pattern was reduced from days down to minutes, while always individuating the test program points where the peak power was maximized.

The rest of the paper is structured as follows: in section II we discuss background concepts about test power evaluation metrics, while section III describes the state-of-the-art power estimation techniques and the related issues. Section IV describes the strategy developed to identify the subset of time points in a functional test pattern that are the best candidates for showing high power consumption. Section V shows the results obtained on two processor cores, the Intel 8051 and the OpenRisc 1200. Section VI draws some conclusions.

II. BACKGROUND

This section gives some basics about power evaluation metrics. Given a set of stimuli S with a certain duration T in seconds, we can split the analysis of the power consumption of S in two components: (i) average power consumption and (ii) maximum of the instantaneous power consumption, also called *peak power consumption*. While elevated average power consumption leads to excessive heat dissipation and hence thermal issues, elevated peak power consumption is the major cause of Power Supply Noise (PSN) (i.e., IR-drop and Ldi/dt events [1]). Excessive PSN can increase gate delays (and hence path delays) in the circuit under test. With an increased delay, some tested paths may be slower than in functional mode of operation though being defect-free. In [3] we quantify that during structural test, peak power is about 24% greater than functional peak power. This may lead to declaring a good chip as faulty, thus leading to manufacturing yield loss [1] [2].

Reflecting the above power components, the power analysis is done considering two power metrics: **cycle average power** and **peak power consumption**. In general, cycle average power refers to the average of instantaneous power consumed by the device during a specified time window. Peak power refers to the highest power value measured during the same time window. The definition of the time window varies according to the stimuli S applied to the device. In the following, we consider two types of stimuli S: (i) functional patterns and (ii) structural test patterns. For each type of stimuli the time window and its meaning are given.

Each at-speed test requires at least two vectors, corresponding to the *launch* and *capture* cycles; hence, structural test patterns power consumption can be evaluated during these two cycles [3]. Launch power is the power consumed during the launch cycle (also called "launch-to-capture" or "test" cycle), which is performed with an at-speed clock cycle. Capture power is the power consumed right after the capture edge, during a time interval also equal to the rated clock period of each considered circuit. Figure 1 shows the time windows in which these power measures are made.

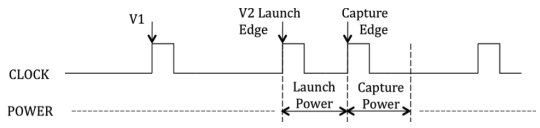

Figure 1. Structural test pattern time windows

A. Functional patterns

In this specific work we consider the case in which the CUT is a processor device. Thus, the considered functional patterns refer to programs executed by the processor. Please note that this is not a limitation, because this choice simply impacts the time window definition.

Before the test program execution, the microprocessor has to execute a small piece of code to ensure the correct initialization of the microprocessor itself.

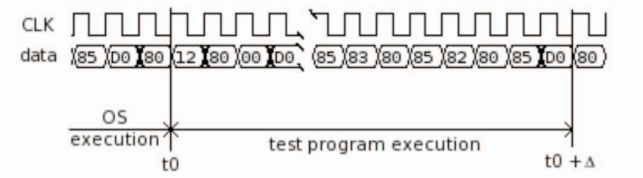

Figure 2. Functional pattern time window

Figure 2 depicts a snapshot of a test program execution waveform. For the sake of readability only the clock signal and the I/O data are reported. After the OS execution, the test program execution starts at time t0 and ends at time t0+Δ. For each test program we consider the time window (t0, t0+Δ) as the period to estimate the cycle average and the peak power.

III. POWER ESTIMATION TECHNIQUES TAXONOMY

In this section we summarize the main techniques devoted to estimate power consumption on processor cores and reports the results of some practical experience over processor cores.

Power estimation techniques may be classified in two approaches: measurement-based and simulation-based. The former requires a processor prototype that supports the possibility of measuring physical properties of the considered circuit. However, regarding processor cores, correct measurements are not easy to obtain since measuring current and voltage of digital circuits must consider a huge range of frequencies where power consumption is inconsistently spread. Thus, measures based on digital multimeters [6] or even oscilloscopes [7] do not provide clear enough information to satisfactorily characterize power consumption of processor cores. The main contribution found in the literature for the power characterization of the instruction set based on physical measurements is given by [6]; in this case, the methodology is based on physical measurements of the current drawn by the processor during the execution of embedded software routines. The authors of [6] analyze both instruction-based costs and inter-instruction costs. The main drawback of those approaches is that they provide useful information for the calculation of average power estimation, but hardly provide any information about peak power estimation.

On the other hand, the most common power estimation techniques that exploit simulation-based approaches may be divided in:

- **Switching Activity (SA):** it corresponds to the toggle activity. Basically we count how many times an object (e.g., the gate output) switches (from 1 to 0 or vice versa) during the stimuli application;
- **Weighted Switching Activity (WSA):** similar to the SA, but the number of switches is weighted depending on the characteristic of the object (i.e., fan out, load capacitance);
- **Transient Analysis (TA):** is based on spice simulation of the stimuli. It provides the current consumption waveform;
- **Post Layout Analysis (PLA):** similar to TA, but it takes into account more parameters, such as RC and power ground grid during the simulation.

Power estimation techniques can be classified depending on both the abstraction level and the representation domain of the CUT.

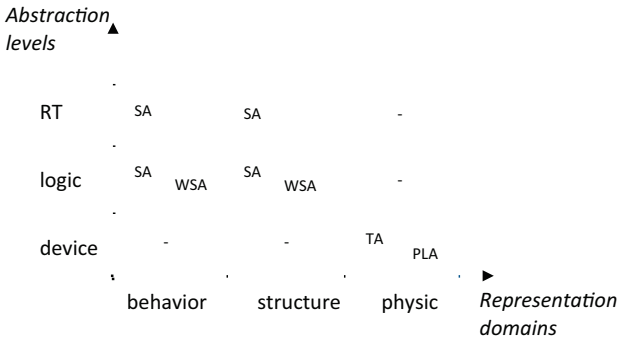

Figure 3. Estimation techniques VS model abstraction

Figure 3 gives the representation matrix. The horizontal axis shows the representation domains (behavioral, structural and physical) while the vertical axis reports the abstraction levels of the CUT (RT, logic and device levels). In the matrix we plotted the above listed estimation approaches. Depending on the position within the matrix the estimation approach can be applied with a different degree of accuracy. For example, the SA at RT level is less precise than the SA at logic (i.e., gate) level. Moreover, some types of estimation approaches make sense only at a specific level. For example, you cannot exploit TA without physical information about your device (i.e., the transistor model). Intuitively, different approaches applied at different levels lead to tradeoff between accuracy and performances.

For the sake of quantifying the accuracy and cost of power estimation methods, we conducted many experiments at different levels of abstraction. The benchmarks used for this study were a couple of microprocessors. As a first result, we report in Table I a qualitative comparison among the power evaluation approaches, together with the estimation of the time required by the performed analysis. This comparison has been done using the same circuit, the 8051 microcontroller (see section V for details), described at different abstraction levels (RTL, logic and device). On this circuit the same set of stimuli (i.e., a given functional program that requires about 1,100 clock cycles to execute) has been applied. As expected, higher level of abstraction leads to faster power evaluation, but achieves lower accuracy. Despite the fact that this qualitative analysis gives somehow well-known and expected results, it is interesting in giving an idea about the differences in terms of required run time (some orders of magnitude) [1].

TABLE I. POWER EVALUATION APPROACHES: QUALITATIVE COMPARISON

Evaluation	Accuracy	Run Time
RTL SA	Very low	1X
Logic SA	Low	2X
WSA	Medium/ High	10X
TA	High	300X
PLA	Very High	1,000X

To provide a visual feedback concerning the accuracy of the measurement and to underline what could be problematic when using high level description to estimate the power consumption properties of a pattern, we report in Figure 4 a significant portion of a pattern and the corresponding results obtained through the various methods listed in Table I. From the top to the bottom of the picture, we plotted first the SA waveform at RT level, then the SA at logic level (i.e., gate level), then we moved to the WSA (obtained by using PrimePower by Synopsys [8]). Finally, the last waveform is the current consumption given by using TA and obtained from Nanosim by Synopsys [4].

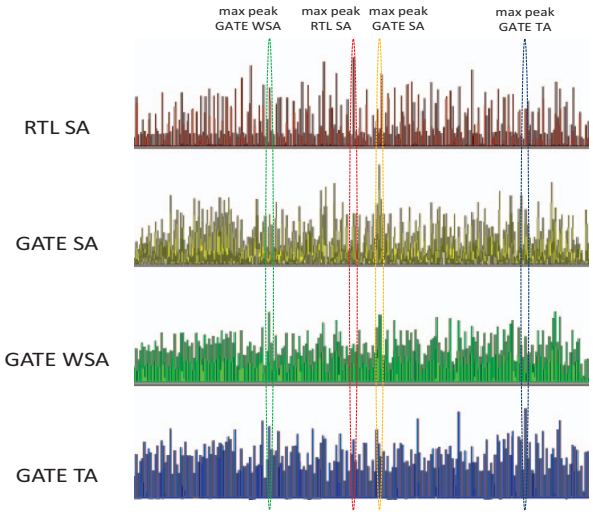

Figure 4. Evaluation approaches quantitative comparison

The first consideration about this figure is that if we look for the cycle average power, the four evaluation approaches converge to a similar value. As a matter of fact, this means that for cycle average power, the SA calculated at the gate level can be considered accurate enough.

The second and crucial consideration relates to the peak power consumption: as shown in the waveforms, the highest peak of the SA (both RTL and gate) may not correspond either to the highest peak of the WSA or to the one of the TA.

The dashed ovals in Figure 4 show the instance in which the highest peak appears for the different estimation methods. As the reader can notice, there is not a correspondence in the maximum peaks for the presented methods.

In particular, this missing correspondence between the peak power and the maximum switching activity is the real motivation of this work. The practical consequence of this observation is that it is impossible to accurately estimate the peak power of a pattern set by only using logic level circuit information.

This assumption makes really difficult to run automatic generation of functional patterns maximizing the actual power consumption of a CPU core [3][5].To overcome such a problematic aspect, we present a novel approach for peak power estimation. This approach still exploits the SA to obtain a quite precise peak power evaluation while keeping a low computational time.

IV. THE PROPOSED APPROACH

In this paper we propose a novel methodology for analyzing the switching activity produced by a functional pattern run by a processor and correlate it with the peak power consumption.

The effort in this direction is motivated by the need of a methodology for screening functional patterns according to their ability to maximize the peak power consumption; this is crucial in a generation flow such as the one proposed in [5] where a function test program is evolved along generations by mixing program chunks according to their evaluation.

Since identifying the maximum peak power appears to be unfeasible by looking only at the switching activity (as described in section III), we tried to relax the requirement in this direction by devising a technique not trying to identify a single point showing the highest peak power consumption in a pattern, but selecting a set of points along the pattern that shows a significant power consumption.

The methodology is quite simple and works as follows:

1. Perform a logic level simulation and record the switching activity, clock cycle by clock cycle
2. Filter the obtained switching activity values by keeping the maximum switching activity within each clock cycle
3. Calculate the average switching activity over the filtered values
4. Select the time points where the switching activity is exceeding the average switching activity by an arbitrary value (i.e. by a percentage as later discussed in the case study) as the candidates for showing the highest peak power.

Figure 5 graphically illustrates the flow.

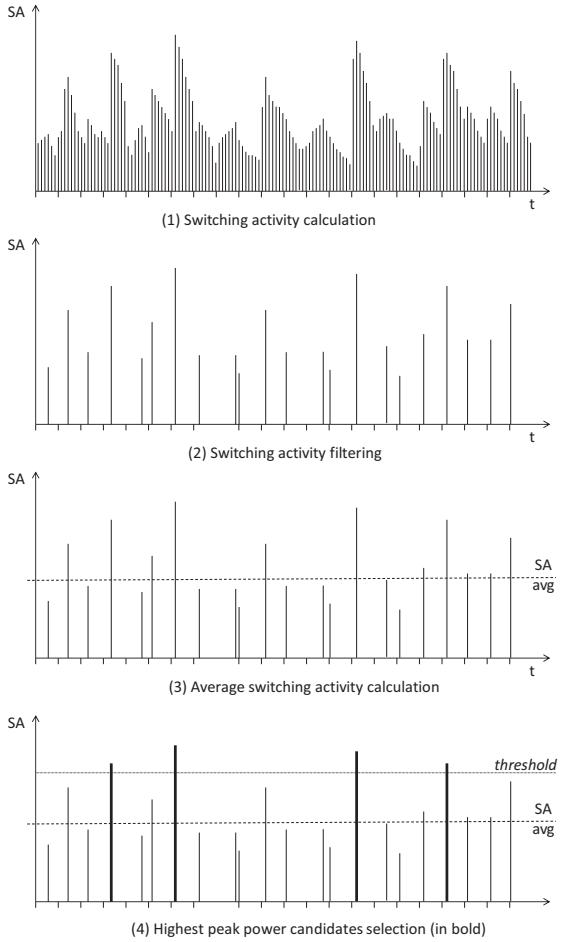

Figure 5. SA in time after every step.

The methodology is quite coarse but it relies on the following strong assumptions:

- It is true that there is no strict correlation among the peak power and the maximum switching activity, but the overall trend of the switching activity is usually indicative of the power consumption value, therefore we expect that the highest power consumption values correspond to the points showing high switching activity
- Also, it is quite usual to observe that most of the time points in a functional pattern produce a low switching activity which also corresponds to low power consumption, therefore it is fair to throw them away.

Concerning the threshold value used to screen out bad candidates, it may be empirically obtained, i.e., by considering a sample set of test programs for evaluation; the major advantage obtained by setting this threshold is just the limitation of the number of potential peak power candidates, which leads to a faster analysis. Section V provides examples for the calculation of this threshold value.

V. CASE STUDIES

In order to validate the proposed methodology, we investigated the peak power characteristics when functional patterns are applied to exercise CPU cores. In particular, we considered the Intel 8051 [10] (non-pipelined) and the OpenRisc [11] (5-stage pipelined) processor cores synthesized on a 65nm industrial technology library. We have also removed the embedded memory cores because they may have a significant impact on the peak power estimation of the CPU core. Characteristics of the processor cores and average evaluation times of SA and TA are reported in Table II. The experiments of both CPU cores were carried out on an Intel Xeon@3.16GHz with 8GB of RAM.

TABLE II. CPU CHARACTERISTICS AND AVERAGE EVALUATION TIMES

	8051	OpenRisc
Combinational gate [#]	~9k	~14k
Sequential gate [#]	~600	~2k
Clock frequency [MHz]	50 MHz	100 MHz
SA evaluation time [s]	~90	~200
WSA evaluation time [s]	~600	~1400
TA evaluation time [s]	~17000	~42000

In both cases we evaluated the power characteristics of functional patterns; the processors are stimulated by executing a program, as in the Software-Based Self-Test methodology. This kind of pattern evaluation is fundamental for the identification of the power limit upper bound.

In both cases, we used the Mentor Graphics ModelSim HDL simulator for gathering switching activity information and the Synopsys PrimePower power evaluator tool for evaluating the weighted switching activity. Finally we used Synopsys Nanosim to execute transient analysis to obtain power measurements. This flow is shown in Figure 6.

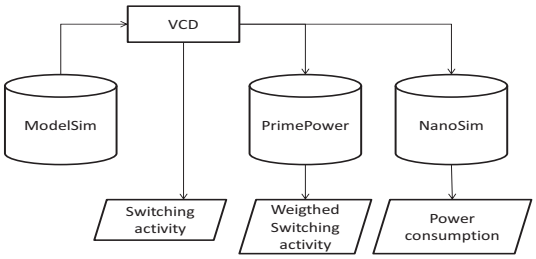

Figure 6. Evaluation flow

The experimental setup shown in Figure 6 was used with 10 test programs for each processor. The programs were selected out of a larger set to be representative of the several power consumption configurations they can activate. They have been sorted in increasing order of peak power.

Table III reports the peak power consumption for every test program. As a first analysis, we evaluated the correlation among the trends of switching activity and power consumption. Table VI shows the values of the best, worst and average correlation index between SA and TA for the 10 test programs. The index was calculated according to Pearson formulation [9] .

TABLE III. PEAK POWER VALUES PER TEST PROGRAM

	Peak power value [mW]	
	8051	OpenRisc
TP1	298.2	913
TP2	383.3	1232
TP3	399.9	1561
TP4	405.8	1651
TP5	410.3	1851
TP6	416.5	2170
TP7	423.3	2450
TP8	432.5	2472
TP9	437.7	2486
TP10	452.5	2534

TABLE IV. CORRELATION INDICES

	case	**Correlation index**
8051	Best	0.8533
	Worst	0.8316
	Average	0.8439
OpenRisc	Best	0.9598
	Worst	0.8802
	Average	0.9429

It can be noticed that the correlation index value is quite high, which means that the trends of switching activity and power consumption are coarsely similar. These results justify the usage of switching activity as the basis for power consumption evaluation techniques.

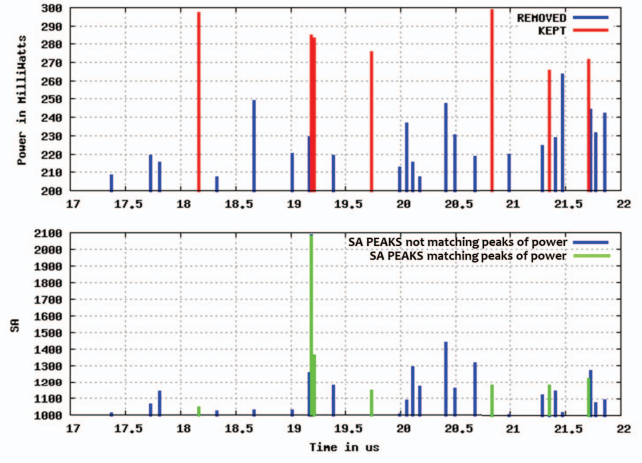

Figure 7. Peak power candidates

On the other side, as already stated in the first part of the paper, in most cases switching activity as it is cannot be used for evaluating peak power consumption.

Figure 7 provides a comparison among the most significant peaks of SA activity and power consumption of a given test program. Please note that we had similar behavior also for the other test programs. The first graph of Figure 7 reports the power consumption (in mW) on the vertical axis. The graph shows the 50 highest peaks of power consumption. The

second graph reports the SA on the vertical axis. It shows the filtered SA peaks (i.e., after the application of the proposed approach). In our experiments, a threshold of 1000 switches within a clock cycle was considered, which is about 30% more than the power consumption average value.

A first comment is related to the fact that the maximum peak power does not correspond to the maximum switching activity peak. A second one is devoted to the results after the application of the proposed approach. Red peaks in the first graph are those matching the filtered SA peaks, in particular these are the green peaks in the second graph. It can be observed that all the highest peaks of power (red peaks of power) always have a correspondent SA peak overcoming the selected threshold (green SA peaks) Table V and VI report the results of the proposed method on both the 8051 and the OpenRisc processors. The tables report in the first column the test program (*TP*) analyzed, the maximum peak power achieved by the program in mW (column 2) and number of switches (column 3). The fourth column shows the correspondence between the 10 highest peaks in SA and TA. Finally, the last column shows whether the maximum peak in SA corresponds or not to the maximum one in TA.

Considering the 8051 processor core, it can be clearly seen from table IV that the proposed method is able to return a reduced set of time point being candidate to show the highest peaks of power (6 to 9 out of the 10 highest peaks). On the other side (Table V), considering the OpenRisc processor, the proposed method is able to identify all the 10 highest peaks in the reported programs.

TABLE V. 8051 TA AND SA PEAKS

TP	Peak Power [mW]	Peak SA	Kept peaks of power (out of 10 best)	Maximum SA & TA peaks at same time?
1	298.2	1179	**1**,2,3,4,6,7,9 (7/10)	No
2	383.3	1744	**1**,2,4,7,8,9 (6/10)	Yes
3	399.9	1228	**1**,2,3,4,6,7,8,9 (8/10)	No
4	405.8	1267	**1**,2,3,4,5,6,8,9 (8/10/)	No
5	410.3	1276	**1**,2,3,4,5,6,8,9 (8/10/)	No
6	416.5	1277	**1**,2,3,4,5,6,7 (7/10)	No
7	423.3	1302	**1**,2,3,4,5,6,7,10 (8/10)	No
8	432.5	1469	**1**,2,4,5,6,7,8,9 (8/10)	No
9	437.7	1584	**1**,2,4,5,6,7,8,9,10 (9/10)	No
10	452.5	1658	**1**,2,3,6,7,8,9,10 (8/10)	No

A major concern of the proposed approach is the selection of the threshold to be used. To determine the threshold value, we first obtain the average power consumption for the 10 test programs applying the TA. Then, we considered a reasonable percentage of the power consumption average. This kind of pre-analysis is slightly expensive, but the obtained result can be further used to evaluate very quickly (an order of magnitude less than WSA analysis) a large set of programs, i.e., during the evolutionary generation of power hungry programs [5].

VI. CONCLUSIONS

We have presented a methodology to estimate functional peak power in processor cores by using the toggle activity metric obtained by executing functional patterns. The method was evaluated on two CPU cores. Results demonstrate that the proposed approach allows resorting to SA for peak power evaluation, thus reducing significantly computational time with respect to alternative solutions. On-going works are mainly devoted to the application of the proposed approach as a fast feedback evaluator of an automatic pattern generator able to create programs increasing peak power consumption.

TABLE VI. OR1200 TA AND SA PEAKS

TP	Peak Power [mW]	Peak SA	Kept peaks of power (out of 10 best)	SA & TA peaks at same time?
1	913	3123	**1** to 10 (10/10)	No
2	1232	5321	**1** to 10 (10/10)	Yes
3	1561	7743	**1** to 10 (10/10)	Yes
4	1651	7456	**1** to 10 (10/10)	Yes
5	1851	7656	**1** to 10 (10/10)	No
6	2170	7635	**1** to 10 (10/10)	No
7	2450	11827	**1** to 10 (10/10)	No
8	2472	11894	**1** to 10 (10/10)	No
9	2486	11945	**1** to 10 (10/10)	Yes
10	2534	12449	**1** to 10 (10/10)	No

VII. REFERENCES

[1] P. Girard, N. Nicolici and X. Wen (editors), Power-Aware Testing and Test Strategies for Low Power Devices, Springer, ISBN 978-1-4419-0927-5, 2009.

[2] J. Saxena, K. M. Butler, V. B. Jayaram, S. Kundu, N. V. Arvind, P. Sreeprakash, and M. Hachinger, "A Case Study of IR-Drop in Structured At-Speed Testing", IEEE Int'l Test Conf., pp. 1098-1104, 2003..

[3] M. Valka, et al., "A Functional Power Evaluation Flow for Defining Test Power Limits during At-Speed Delay Testing", IEEE European Test Symposium, pp. 153-158, 2011.

[4] Synopsys Inc., NanoSim™, User Guide 2009.

[5] A. Calimera, et al., "Generating power-hungry test programs for power-aware validation of pipelined processors," in Proc. of the 23rd symposium on Integrated circuits and system design, pp 61-66, September 2010.

[6] V. Tiwary, S. Malik and A. Wolfe, "Power analysis of embedded software: A first step towards software power minimization", IEEE Transaction on VLSI systems, vol. 2, no. 4, pp. 431-445, Dec. 1994

[7] J. Russel and M. Jacone, "Software power estimation and optimization for high perfomance, 32-bit embedded processors", International Conference on Computer Design, October 1998, pp. 328-333

[8] Synopsys Inc., PrimeTilme-PX™, User Guide 2011

[9] A. Buda and A.Jarynowski (2010) Life-time of correlations and its applications vol.1, Wydawnictwo Niezalezne: 5–21, December 2010, ISBN 978-83-915272-9-0

[10] http://www.oreganosystems.at

[11] http://opencores.org/