Black-Box Leakage Power Modeling for Cell Library and SRAM Compiler

[1]Chun-Kai Tseng [1]Shi-Yu Huang [1]Chia-Chien Weng [2]Shan-Chien Fang [3]Ji-Jan Chen
[1]EE Dept., National Tsing Hua University, Taiwan
[2]TinnoTek Inc., Taiwan
[3]Information and Communications Research Lab., Industrial Technology Research Institute, Taiwan

Abstract [1] **—In this paper, we present an automatic leakage power modeling method for standard cell library as well as SRAM compiler. For this problem, there are two major challenges – (1) the high sensitivity of leakage power to the temperature (e.g., the leakage power of an inverter can be different by 19.28X when temperature rises from 25°C to 100°C in 90nm technology), and (2) the large number of models to be built (e.g., there could be 80,835 SRAM macros supported by an SRAM compiler). Our method achieves high accuracy efficiently by two formula-based prediction techniques. First of all, we incorporate a quick segmented exponential interpolation scheme to take into account the effects of the temperature. Secondly, we use a MUX-oriented linear extrapolation scheme, which is so accurate that it allows us to build the leakage power models for all SRAM macros based on linear regression using only the simulation results of 9 small-sized SRAM macros. Experimental results show that this method is not only accurate but also highly efficient.**

Index Terms—**Leakage Power Modeling, Leakage Power Estimation, Standard Cell Library, SRAM Compiler**

I. INTRODUCTION

With IC technology scaling, the leakage power consumed by transistors could grow dramatically and even worse it could grow exponentially with the temperature. As a result, the leakage power that may seem to be negligible under the room temperature could contribute to more than 50% of the total power in advanced technology when an IC operates at a high temperature [2][13]. Thus, power estimation needs to include not just the dynamic power, but also the temperature dependent leakage power. Thus, for each standard cell and SRAM compiler-generated macro, we need to build a temperature-aware leakage power model.

Some methods have been reported to estimate the full chip leakage. For example, the authors of [5] used a linear regression model to estimate full-chip leakage mostly based on the information of the gate count. This method is quite efficient. Yet, it may not be very accurate since it is known that the leakage of a cell has a stronger-than-linear dependency on the temperature. The authors in [1] take into account the effects of the input patterns on the leakage by analyzing the transistor stacks in a standard cell. This method successfully simplifies the complexity of the leakage power modeling by categorizing the originally 2^n-entry model of an n-input cell into a much more compact model with only a small number of entries. In [10], the model presented in [1] was further extended by the neural network to include the effects of the operating temperature and the supply voltage. A minor drawback of this method is that it requires the detailed information of each standard cell. In general, a black-box method (which requires no information of a cell's implementation) is often desirable since it is applicable from one standard cell to another. In [11], Zhang et al. developed a method called *HotLeakage*, which analyzes the temperature-dependent cache leakage power systematically. Regarding the power estimation incorporated these leakage models, a method was proposed in [7].

It is known that power and thermal could affect each other in an interactive way. Some previous works [3][4][6][9] have tried to analyze such power-thermal co-estimation mechanism so as to predict two essential quantities that could affect an IC's reliability – (1) the steady-state operating temperature when running a specific application, and (2) the realistic thermal-aware power consumption – the one that is truly measured from an IC operating on a circuit board. In [12], a quick power-thermal co-simulation engine is also used to do the thermal-aware floorplaning.

The above power-thermal co-estimation methods require accurate thermal-aware leakage power models as the foundation. In the near future, this will be inevitable responsibility of the standard cell builders as well as the IP providers (such as SRAM compiler provider). For a given standard cell or an IP (e.g., SRAM macro), one can use SPICE simulation to build this leakage model as a two-dimensional table in terms of two factors – (1) the operating temperature and (2) the input pattern. Using traditional wisdom, one can perform sampling on the temperature and use interpolation to predict the leakage powers of those non-sampled temperatures, while enumerating all possible input patterns since the total number of input patterns is mostly affordable. This baseline approach sounds applicable since leakage power modeling for standard-cell library is a one-time effort. However, the efficiency is still a major concern, considering that person who takes charge of the leakage power modeling may need to perform SPICE simulation many times for each cell, and for many cells in a library. A better approach could deliver the solution in a couple of hours, while a naïve one could take days to complete. The key problem of this baseline approach is how one can do the temperature sampling wisely without hurting the accuracy. This problem is also linked to the underlying formula to be used for the interpolation. After in-depth investigation, we present in this paper an efficient and accurate scheme – i.e., segmented exponential modeling – to be

[1] This work was supported in part by *National Science Council (NSC) of Taiwan under grant NSC-98-2220-E-007-005*.
978-3-9810801-7-9/DATE11/©2011 EDAA

described in detail later.

As for SRAM complier, we need to address another two issues – i.e., (1) the large number of configurations to be modeled (which is as high as 80,835 in our example), and (2) the simulation time explosion for large-sized SRAM macros (e.g., one with 256Kbits). This is a problem one will not encounter when building the leakage power models for standard cells.

To avoid the simulation of large-sized SRAM macro, a better approach will take only small-sized SRAM macros for leakage power simulation using SPICE (or other quick transistor-level simulation tools). Based on the collected leakage power information, an extrapolation scheme (not just interpolation) is further utilized to predict those of the larger SRAM macros. Certainly, the accuracy of such extrapolation strongly depends on the underlying formula. A better formula should be able to capture the relationship between the leakage power and the structural parameter of the SRAM (such as the number of bits in a word, the total number of wordlines, and the so-called MUX width, etc.). Our experience shows that a better formula can also reduce the required number of sample configurations, and thereby reducing the modeling time dramatically. In this paper, we also reveal such an accurate extrapolation scheme based on MUX-oriented linear formula to be discussed in detail later.

In summary, our work makes the following contributions:

(1) Our method is completely a black-box one, which does not require the knowledge of the implementation of the standard cell or the SRAM compiler.

(2) Our method reduces the required number of temperature samples at which the (quick) SPICE simulation needs to be performed by finding a segmented exponential formula that fits the leakage power nicely. The benefits include much faster modeling time and high accuracy.

(3) To our knowledge, this is the first automatic black-box leakage power modeling tool for SRAM compilers. It is generic since the MUX-oriented linear model is built in terms of the functional parameters of SRAM macros. The high accuracy this model enable the extrapolation scheme so that we can skip the SPICE simulation of large-sized SRAM macros, and thereby making the modeling process tractable within reasonable CPU time.

The rest of the paper is organized as follows. Section Ⅱ and Ⅲ present our power modeling techniques for standard cell libraries and SRAM compilers, respectively. Section Ⅳ shows the experimental results and Section Ⅴ concludes.

II. LEAKAGE POWER MODEL OF A STANDARD CELL LIBRARY

A. Temperature-Aware Leakage Power Modeling

The purpose of leakage power modeling for a given standard cell library is to approximate the leakage power of every cell under every possible input pattern as a table or a number of formula or a combination of both. Considering the effect of the input patterns alone, if the given cell library has N cells and the i^{th} cell has M_i inputs, then the total number of entries would be $\sum_{i=1}^{N} 2^{M_i}$ in total. Because of the large amounts of models that need to be generated, there are many ways that have been proposed to cope with the complexity issue. One way is to model the average leakage power of a cell across all its input patterns. This method can reduce the total number of models, but it assumes that each input pattern is equally likely to occur. This assumption is overly simple since the probability of occurrence of an input pattern for a cell depends on the circuit operation. Besides, according to *SPICE* simulation results in Fig. 1, the leakage power of a two-input NAND cell varies from one input pattern to another. The leakage power is 0.272 nW and 4.695 nW at 100°C when we apply all-'0' pattern and all-'1' pattern, respectively. It can be calculated that the leakage power could have a 17X difference between these two input patterns. Thus, our method is to model the leakage power under every input pattern for each standard cell.

B. Temperature-Dependent Characteristic Formula

As shown in figure 1, the characteristic curve of the leakage power versus the operating temperature is an exponential-like shape (under one specific input pattern). As a result, using linear regression to model the relationship between the leakage power and the temperature may not be accurate enough [8]. Therefore, we choose to use the following *exponential regression* method instead:

$$P_{leak} = Ae^{B \bullet TEMP} \qquad (1)$$

Where P_{leak} represents the leakage power, TEMP represents the temperature and A, B are two key weighting factors to be derived from the leakage power data under a number of sampled temperatures.

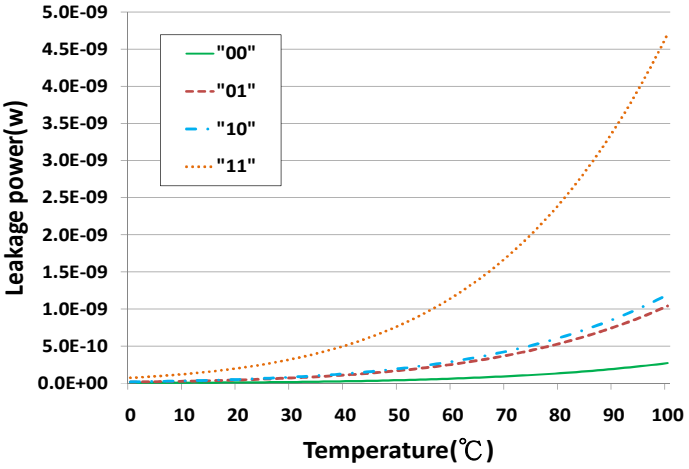

Fig. 1: Leakage power for each input pattern of a 2-input NAND cell using 90nm CMOS process.

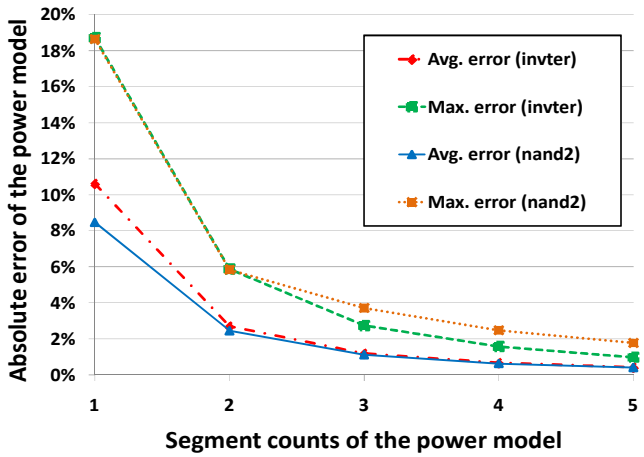

Fig. 2: Error of piecewise exponential model in different segment counts.

If the target temperature range is from 0°C to 100°C, the result of exponential regression has an average absolute error of about 10% across all input patterns for a two-input NAND cell, regardless of the total number of temperature samples (i.e. the temperatures at which leakage power values are derived by SPICE simulation) used for the regression. *In order to improve the accuracy of the models, we found that dividing the target range into a number of segments of exponential models is better than using more regression samples*. Fig. 2 shows the average and the maximum absolute error of such a segmented exponential model, considering the effects of different segment counts for two basic cells. In this experiment, we use only three uniformly distributed regression samples per segment; which is the smallest number of samples to represent an exponential curve. Thus, we can reduce the *SPICE* run-time significantly during the leakage power modeling. As shown in Fig. 2, the average error and the maximum error of the model decrease quickly as the number of segments increases. As the segment counts equal four, the average absolute error has dropped down to 0.63% for a NAND2 cell, and 0.67% for an inverter cell. When the number of segments is larger than 4, the characterization complexity increases unnecessarily without noticeable increase in the accuracy. Based on these simulation results, we thus use 4 as the number of segments in our modeling methodology to retain enough accuracy within reasonable modeling time.

Further, we compare our models with the *HSPICE* simulation results. The Fig. 3 shows the leakage power consumption of a NAND2 cell within a temperature range of [0°C, 100°C]. No matter what input pattern is applied, the leakage power predicted by our model matches the *HSPICE* simulation results very well throughout the entire temperature range.

III. Leakage Power Model of An SRAM Compiler

A. Basic Idea

In this work, our goal is to model the leakage power of memory macros in various configurations (i.e., word depths, word-widths, and IOs) and at different temperatures. If we take Artisan memory compiler [14] for example, there are in total 80,835 memory configurations. Unlike the standard cell library, it is impossible to do the power modeling through SPICE simulation on each configuration. We need an extrapolation scheme that can predict the leakage power of those larger configurations using only small configurations. We found that SRAM macros with different MUX-widths (i.e., the number of bitline pairs sharing an IO line) tend to have quite different power consumption behavior. Thus, a MUX-oriented model is more appropriate. It is like we cannot model the leakage power of all SRAM macros in one formula very accurately, but we found that we are able to model them in several MUX-width-specific formulae. By doing so, only a few small memory configurations are adequate to serve as the training set for deriving accurate leakage powers for all the other macros. This method integrated with the techniques discussed in previous Section (i.e., temperature sampling plus segmented exponential regression to take into account the temperature effect) could provide accurate leakage power for every SRAM macro at any temperature within our target range.

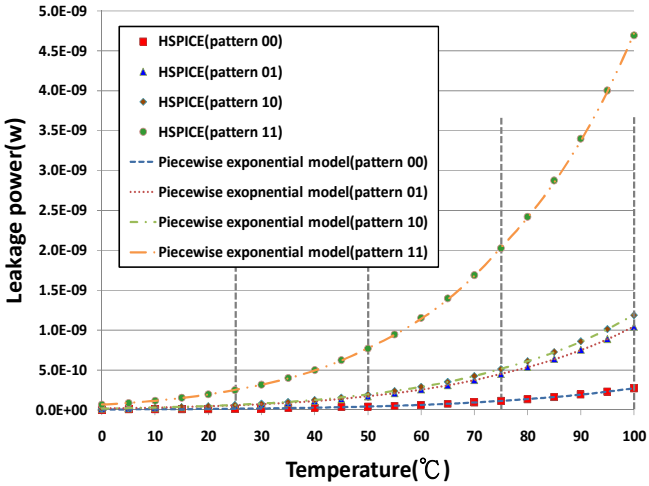

Fig. 3: Leakage power for *HSPICE* and segmented exponential models using 90nm CMOS process for a NAND2 cell.

B. Characteristic Formula for SRAM Compiler

In this sub-section, we discuss the detail of the MUX-oriented leakage power model we propose for SRAM macros generated by SRAM-compiler.

1) MUX-Oriented Power Model

As mentioned previously, approaches incorporated a single power formula for all configurations of a memory compiler did not lead to highly accurate results. Thus, we split all SRAM configurations into a number of major categories based on their MUX-width; each category will then have its own power formula. By doing so, the power formula will match the simulated results more closely. The power model is defined as follow:

For a certain MUX-width M_w:

$$\text{Power} = C_0 + C_1W + C_2B + C_3W \cdot B \qquad (2)$$

Where C_0, C_1, C_2, C_3 are weighting factors, W is the number of words, B is the number of bits and WxB is the memory array size.

We analyze the relationship of the predicted leakage power using the MUX-oriented power and actual leakage power through SPICE simulation in the training set at different temperatures as shown in Fig. 4 and Fig. 5. The power data match their ideal values closely at both 25°C and 100°C.

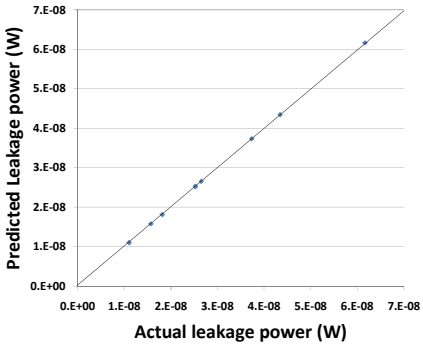

Fig. 4: Predicted leakage power based on MUX-oriented model vs. actual leakage power of SRAM macros in the training set at 25°C

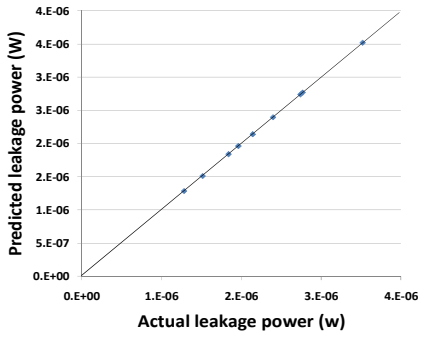

Fig. 5: Predicted leakage power based on MUX-oriented model vs. actual leakage power of memories in the training set at 100°C

2) Segmented Exponential Model for SRAM Macros

So far, we can derive the leakage power for different SRAM configurations at some specific temperature by using the MUX-oriented power model. But we do not have the ability yet to predict the leakage power at the temperatures which we have not generate its own MUX-oriented model. In some sense, the leakage power modeling for a standard cell library is a one-dimensional prediction problem (since we only predict the effect of the temperature), while the modeling for SRAM compiler is a two-dimensional prediction problem, (in which we consider the effects of both the SRAM configuration and the temperature). Therefore, we need to apply the concept of *segmented exponential model* previously used for the standard cell library again to SRAM macros. The only difference is how *we produce the leakage power samples over a wide range of temperatures. In this 2-dimensional prediction scheme, we perform the MUX-oriented modeling for varying configurations for a number of selected temperatures first. Then, we fill in the leakage powers for each SRAM configuration under those un-sampled temperatures by segmented exponential regression.*

C. Deriving Leakage Power of a Memory Macro

The overall flow for leakage power modeling for an SRAM compiler is a little bit more involved as shown in Fig. 6. It has two phases – the off-line phase and on-line phase. As mentioned previously, two types of formulae will be used, namely (1) *MUX-oriented model* (considering the effect of the configuration) and (2) *Segmented exponential model* (considering the effect of temperature). Building MUX-oriented power is an off-line procedure, while building segmented exponential model is an on-line procedure which is performed dynamically on demand.

In the off-line phase, we first run the SPICE simulation of those SRAM macros in the training set for all sampled temperatures. After that, we run linear regression to create the MUX-oriented power model at every sampled temperature.

In the on-line phase, once a SRAM macro is given with its memory configuration and the operating temperature, the leakage power can be calculated in three steps: (1) We calculate the leakage power data of this configuration at those sampled temperatures by applying the MUX-oriented formulae. (2) Then from the above leakage power data, we dynamically produce the segmented exponential model specifically for this SRAM configuration by running regression. (3) Finally, we apply the obtained the segmented exponential model at the specific operating temperature to derive its leakage power value.

Fig. 7 shows the predicted leakage power (i.e., the leakage power predicted by our method) and simulated power through the *NanoSim* simulation (which is used as a reference value) for two configurations W64B12M4 and W64B32M4 at different temperatures. The power data is close to the *NanoSim* simulation values.

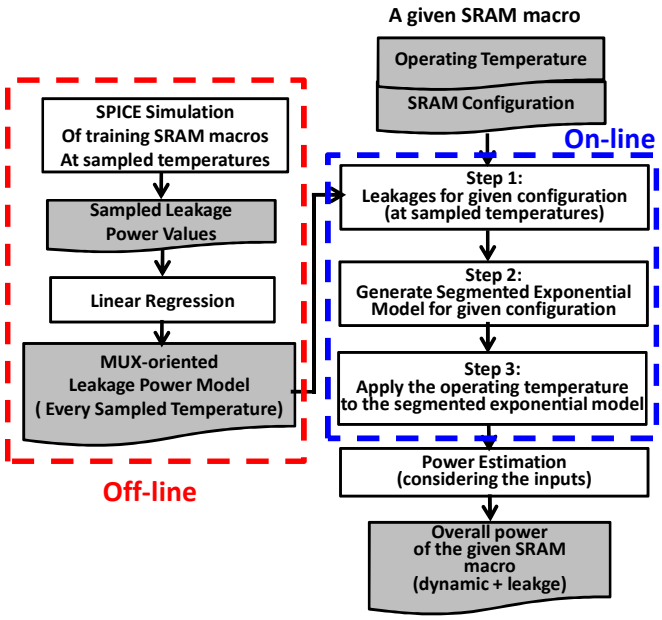

Fig. 6: Overall flow of our 2-dimensional leakage power modeling for SRAM compiler.

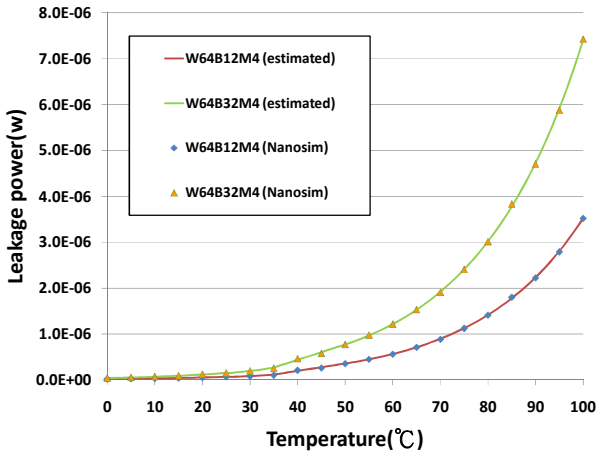

Fig. 7: Leakage power for *Nanosim* and our model using 0.18 um process for W64B12M4 and W64B32M4

IV. EXPERIMENTAL RESULTS

We have implemented our method as a tool in C++ language and Perl 5.8. Our tool links with the commercial tools, *Calibre* from *Mentor Graphics* for post-layout netlist extraction, *Hspice* and *NanoSim* from *Synopsys* for power simulation.

A. Modeling Accuracy of Basic Cells

To verify the accuracy of our models for standard cell library, we built a total of eight basic cells. Table 1 shows the average absolute error and the maximum absolute error of each cell. It can be seen that the average error is 0.64% and average maximum error is 2.20% across all cells.

Table 1: Results of the modeling error for different cells using 90nm process.

Gate	Average Absolute Error (%)	Maximum Absolute Error (%)
INV	0.67%	1.57%
AND2	0.60%	1.38%
NAND2	0.63%	2.53%
NOR2	0.63%	1.38%
XOR2	0.62%	1.35%
HA	0.62%	1.40%
LATCH	0.60%	1.35%
FF	0.73%	6.66%
Average	0.64%	2.20%

B. Modeling Accuracy of SRAM Macros

For SRAM macros generated by a SRAM compiler, we choose 9 small-sized memory macros with different configurations with the MUX-width equaling 4 as our training set. Besides, we generate another 5 medium-sized memory macros as our validation set. As shown in Table 2 and Table 3, the average absolute error is 4.87% and 3.89% for training set and validation set, respectively.

Table 2: Results of the modeling error for the training set using 180nm process.

Memory Configuration	Average Absolute Error (%)	Leakage Power (W)	
		25℃	100℃
W16B4M4	6.00%	1.10E-08W	1.28E-06W
W16B8M4	5.43%	1.81E-08W	1.84E-06W
W16B12M4	5.16%	2.52E-08W	2.40E-06W
W32B4M4	5.65%	1.57E-08W	1.51E-06W
W32B8M4	4.85%	2.65E-08W	2.14E-06W
W32B12M4	4.50%	3.73E-08W	2.77E-06W
W64B4M4	4.82%	2.53E-08W	1.96E-06W
W64B8M4	3.88%	4.35E-08W	2.74E-06W
W64B12M4	3.47%	6.16E-08W	3.52E-06W
Average	4.87%	-	-

Table 3: Results of the modeling error for the validation set using 180nm process.

Memory Configuration	Average Absolute Error (%)	Leakage Power (W)	
		25℃	100℃
W64B32M4	2.84%	1.53E-07W	7.43E-06W
W128B16M4	3.28%	1.43E-07W	5.87E-06W
W128B32M4	2.53%	2.75E-07W	1.01E-05W
W512B8M4	7.17%	2.76E-07W	9.69E-06W
W256B32M4	2.76%	5.23E-07W	1.54E-05W
W512B16M4	4.73%	5.19E-07W	1.52E-05W
Average	3.89%	-	-

C. Modeling Time

Our experiments are performed on a PC-based Linux workstation with Intel Xeon 3.0GHz x 8 CPU and 64GB RAM. We use *NanoSim* as the underlying transistor-level simulation engine. Consider the execution of our tool on the mini cell library we built. It takes about 2 minutes and 13.32 seconds overall. On the average, it takes about only 16.67 seconds for each basic cell.

As for the execution for SRAM compiler, the overall runtime was 1 hour, 6 minutes, and 33 seconds. Most of the time is spent on the so-called power characterization process – i.e., running the *NanoSim* simulation for the SRAM macros in training set at sampled temperatures. Compared with the power characterization time, the time spent on the regression of the segmented exponential modeling is negligible. Overall, we include 9 SRAM configurations in the training set for each MUX-width. Since it is impossible to run the *NanoSim* simulation for some large configuration, extrapolation is the only way to go and we believe that about 1 hour of modeling time is very efficient.

D. *Temperature-Aware Power Estimation*

To show the effect of temperature on leakage power, we use the Advanced Encryption Standard (AES) as a test case. The AES is synthesized using 90nm low-Vth standard cell library. In this design, a single-port SRAM macro with 256 128-bit words is used. The MUX-width is 4. As shown in Fig. 8, the leakage power of the SRAM macro in this design grows drastically when the temperature rises. The leakage power is 1212.99 uW at 25°C, and increases to 7992.47 uW at 100°C, respectively.

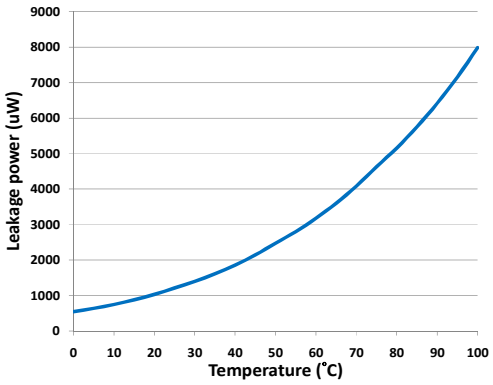

Fig. 8: Estimated leakage power at different temperatures for an SRAM macro in a test case called AES.

V. CONCLUSION

Temperature-aware leakage power modeling for cell library and SRAM compiler is not an easy task if both accuracy and efficiency are required. Especially for SRAM compiler, both the effects of the configuration and temperature need to be considered simultaneously, leading to a two-dimensional prediction problem. We have demonstrated that several techniques, such as temperature sampling plus segmented exponential regression, MUX-oriented linear regression and extrapolation for large SRAM macros, etc. can jointly deliver a fast and accurate solution. The proposed method is very fast in that it takes only 16.67 seconds to handle one standard cell, and 1hr:6min:33s to handle 32,385 SRAM macros supported by an in-house SARM compiler. The average absolute error for the 8 cells we built in a mini cell library is only 0.64%, and that for an SRAM compiler is 4.87% for the 9 training macros, and 3.89% for a set of medium-sized validation set, respectively. To our knowledge, this is the first black-box leakage power modeling method for SRAM compilers. We have integrated this leakage power modeling feature into an in-house power estimation tools, i.e., *PowerMixer*. With this feature, one can not only do the total power estimation more accurately by considering the effect of the operating temperature of an IC design, but also investigate how power and temperature react to one another and under what condition the notorious thermal runaway could have happened.

ACKNOWLEDGMENT

The authors would like to thank *CIC* (Chip Implementation Center), Taiwan, for their help in providing the needed commercial tools, including *Calibre*, *Hspice* and *NanoSim* used in our experiments. We also thank *ITRI* (Industrial Technology Research Institute), Taiwan, for their valuable comments and discussion.

REFERENCES

[1] Z.-P. Chen, M. Johnson, L.-Q. Wei, and W. Roy, "Estimation of Standby Leakage Power in CMOS Circuit Considering Accurate Modeling of Transistor Stacks," *Proc. of Int'l Symp. on Low Power Electronics and Design,* pp. 239- 244, Aug 1998.

[2] A.S. Grov, "Changing Vetors of Moore's Law—Keynote Speech," *Int'l Electron Devices Meeting*, Dec. 2002.

[3] A. Gupta, N. Dutt, F. J. Kurdahi, K. S. Khouri, and M. S. Abadir, "STEFAL: A System Level Temperature- and Floorplan-Aware Leakage Power Estimator for SoCs," *Proc. of VLSI Design*, pp.559-564, Jan. 2007.

[4] Lei He, Weiping Liao, and Stan, M.R., "System Level Leakage Reduction Considering the Interdependence of Temperature and Leakage," *Proc. of Design Automation Conference,* pp. 12- 17, 2004.

[5] R. Kumar and C.P. Ravikumar, "Leakage Power Estimation for Deep Submicron Circuits in an ASIC Design Environment," *Proc. of Asian and South Pacific Design Automation*, pp.45-50, 2002.

[6] W. Liao, F. Li, and L. He, "Microarchitecture Level Power and Thermal Simulation Considering Temperature Dependent Leakage Model," *Proc.of Low Power Electronics and Design,* pp. 211- 216, Aug. 2003.

[7] Y.-S. Lin, and D. Sylvester, "Runtime Leakage Power Estimation Technique for Combinational Circuits," *Proc. of Asian and South Pacific Design Automation*, pp.660-665, Jan. 2007.

[8] Y.-P. Liu, R.P. Dick, S. Li, and H.-Z. Yang, "Accurate Temperature-Dependent Integrated Circuit Leakage Power Estimation is Easy," *Proc. of Design, Automation, and Test in Europe Conference,* pp.1-6, April 2007.

[9] H. Su, F. Liu, A. Devgan, E. Acar, and S. Nassif, "Full Chip Leakage-Estimation Considering Power Supply and Temperature Variations," *Proc. of Low Power Electronics and Design*, pp. 78- 83, Aug. 2003.

[10] J. Viraraghavan, B. P. Das, and B. Amrutur, "Voltage and Temperature Scalable Standard Cell Leakage Models Based on Stacks for Statistical Leakage Characterization," *Proc. of 21st Int'l Conf. on VLSI Design*, pp.667-672, Jan. 2008.

[11] Y. Zhang, D. Parikh, K. Sankaranarayanan, K. Skadron, and M. Stan, "Hotleakage: A Temperature-Aware Model of Subthreshold and Gate Leakage for Architects," Technical Report TR-CS-2003-05, Univ. of Virginia, Dept. of Computer Science, Mar. 2003.

[12] P. Zhou, Y. Ma, Q. Zhou, and X. Hong, "Thermal Effects with Leakage Power Considered in 2D/3D Floorplanning," *Proc. of Computer-Aided Design and Computer Graphics,* pp.338-343, Oct. 2007.

[13] Intel Corp. http://software.intel.com/en-us/articles/gigascale-integration-challenges-and-opportunities.

[14] "Artisan Standard Library SRAM Generator User Manual," Artisan Components, 2004.