A Hybrid Power Model for RTL Power Estimation

Yi-Min Jiang[‡], Shi-Yu Huang[§], Kwang-Ting Cheng[‡], Deborah C. Wang[*], ChingYen Ho[*]

[‡]Dept. of Electrical & Computer Engineering
University of California, Santa Barbara
CA 93106, USA
Tel: +1-805-893-8847
Fax: +1-805-893-3262
e-mail: {jym, timcheng}@bigbend.ece.ucsb.edu

[§]National Semiconductor Corp.
2900 Semiconductor Drive
Santa Clara, CA 95052, USA
Tel: +1-408-721-3251
Fax: +1-408-773-0978
e-mail: syhuang@berlioz.nsc.com

[*]LSI Logic Corp.
1501 McCarthy Blvd.
Milpitas, CA 95035, USA
Tel: +1-408-954-4416
e-mail: dwang@lsil.com

Abstract

We propose a hybrid power model for estimating the power dissipation of a design at the RT-level. This new model combines the advantages of both RT-level and gate-level approaches. We investigate the relationship between steady-state transition power and overall power dissipation. We observe that, statistically, two input sequences causing similar amount of *steady-state* transitions will exhibit similar overall power dissipation for an RTL module. Based on this observation, we propose a method to construct a hybrid power model for RTL modules. We further propose a hierarchical power estimation method for estimating the power dissipation of data-path consisting of RTL modules. Experimental results show that, for full-chip power estimation, the estimation time of the technique based on our power models is on average 275 times faster than directly running a commercial transistor-level power simulator, and the errors are less than 6% as compared to the transistor-level power simulation results.

1. Introduction

With the advent of portable and high density devices, the power dissipation of VLSI designs has become a critical concern. Higher integration and higher speed result in significant power dissipation, which makes heat dissipation and packaging more serious problems. Therefore, the reduction of power dissipation is an important issue in modern circuit design. To reduce power dissipation during synthesis, accurate power estimation at a higher level of abstraction is essential because it provides the designers with an early measure of power dissipation and allows exploration of various design trade-offs in the solution space. In the modern VLSI design process, the register-transfer level (RTL) description has become a common entry point. Design decisions made at this level could have dramatic impact on the total power budget.

For CMOS circuits, the overall power dissipation consists of four components: power consumed by (1) steady-state transition current, (2) glitch current, (3) short-circuit current, and (4) static current. The steady-state transition current occurs when the input stimuli causes a node to change its stable state (from high to low, or low to high). This component is recognized as an important factor in power dissipation, and can be calculated by a logic simulator using the zero-delay model. Glitch current occurs when the load capacitance is charged or discharged before the node reaches its stable state. This component is much more difficult to compute because it is very sensitive to the real circuit delay and partial V_{dd}-ground swing. Short-circuit current occurs whenever a path from V_{dd} to ground is conducted in a device. Static current is caused by leakage on a device. Throughout this paper, we will use the term *steady-state power* to refer to the average power dissipation by component 1, the steady-state transition current. And we will use the term *hazardous power* to refer to the average power dissipated by components 2, 3, and 4. The term *total power* refers to the sum of the steady-state power and hazardous power.

We focus on the problem of power estimation for RTL designs with a given long stimuli. In general, for a problem of this complexity, existing RT-level techniques produce lower accuracy, while existing gate- and transistor-level techniques suffer long run time. In this paper, we propose a hybrid power model that combines the accuracy of gate- and transistor-level models with the efficiency of RT-level models. Our methodology takes both steady-state power and hazardous power, i.e., all possible sources of power dissipation, into account at the RT-level. The main idea in our methodology is based on an important observation. For a given RTL module, empirically we observe that sufficiently long input sequences which produce similar steady-state power will exhibit similar total power. This important observation, in spite of its simplicity, leads us to our power estimation methodology. We will more rigorously analyze the observation in Section 3.

Our methodology consists of a pre-processing stage and an actual estimation stage. The pre-processing stage is a characterization step for each RTL module. For a given RTL module *m*, during this characterization process we establish a function, called *cross-coefficient function*, that one-to-one maps the steady-state power to the total power of module *m*. The cross-coefficient function of each module is characterized only once, and can be repeatedly used in the actual estimation stage when we simulate the whole RTL design with the user-provided long sequence. Note that because of the establishment of the cross-coefficient function for each module, we only need to perform zero-delay simulation in the second process while all sources of power dissipation are taken into account.

We propose a hierarchical power estimation technique based on the above methodology. Essentially our methodology achieves efficiency by performing only zero-delay simu-

lation, and meanwhile achieves transistor-level accuracy through the tight establishment of the cross-coefficient functions. Furthermore, for large parameterized modules, i.e., modules with the same type of functionality but different width of input/output bits, we also develop an interpolation technique so that the module characterization process can be simplified.

The rest of this paper is organized as follows. Section 2 gives review on related research works. Section 3 presents an experiment that validates our important observation. Section 4 describes the module characterization process. Section 5 presents our interpolation technique for parameterized modules. Section 6 gives the complete flow of our technique. Section 7 presents our experimental results and Section 8 draws some conclusions.

2. Related Review

Several power models have been proposed for RTL modules [5][7][9] and logic gates [1][8]. In [7], an RT-level model, called Dual Bit Type, is used to account for the correlated activities at the most significant bits, and the random activities at the least significant bits. Based on these statistics, the capacitive coefficients for each module are derived, and then used for power estimation. Mehta *et al.* [9] have presented a clustering-based power modeling technique which partitions all possible input patterns of an RTL modules into several clusters in a manner that the patterns in the same cluster produce similar power dissipation. Power dissipation for each pattern is then estimated by looking up the mean power in the corresponding cluster. Hsieh *et al.* [5] propose two RT-level power model equations, which are evaluated for all input patterns to obtain the power estimates. A regression estimator is applied to improve the estimation accuracy.

The gate-level power models are addressed in [1][8]. In gate-level approaches, circuit-level power simulation is performed to build power models for logic gates, and the models are then used to estimate the power dissipation during comprehensive gate-level simulation. Lin *et al.* [8] propose a finite-state machine to model the internal charge status of logic gates, and power dissipation during input transitions is represented by weights associated with the state transition of the FSM. Bogliolo *et al.* [1] propose a gate-level power model based on the charge and discharge of load capacitances, and the flow of short circuit current. Several power effects, such as charge sharing, short circuit current, and misaligned multiple input transition are taken into account.

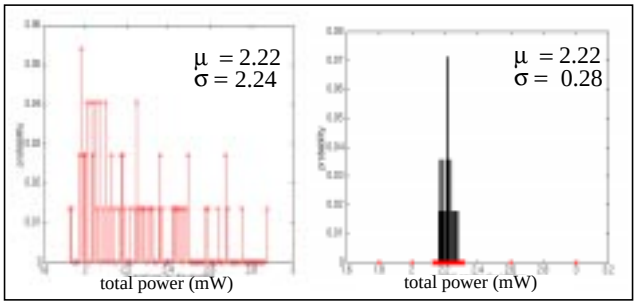
Recently, several techniques have been proposed for power estimation by investigating the relationship between the gate-level and the transistor-level power estimates [6][11]. Using statistical approaches, they simulate only a small set of sequences and then use the results to approximate the power estimate of a much longer input sequence. Although promising results were reported for randomly generated input sequences, these approaches may not be suitable for functional sequences with high spatial and temporal correlation.

3. A Validation Experiment

As stated in Section 1, our methodology is based on an important observation. Roughly speaking, for a given RTL module, empirically we observe that sufficiently long input sequences which produce similar steady-state power will exhibit similar total power. We try to explain, analyze, and validate this observation in this section.

Given an RTL module, for all the input sequences that produce a fixed steady-state power, we believe that the hazardous power corresponding to an input sequence has the behavior of a random variable. Furthermore, among all these input sequences that produce a fixed steady-state power, longer sequences tend to have smaller variance than shorter sequence. As an example, given a 3-input logic network, assume sequences s_1={101, 110}, s_2={111, 101}, s_3={110, 011, 000, 111, 101, ...}, s_4={010, 101, 111, 010, 011, ...}, ..., all exhibit the same steady-state power (say, obtained by zero-delay logic simulator). We believe that the hazardous power produced by all these sequences has the behavior of a random variable, and longer sequences, such as s_3 and s_4, have a smaller variance than shorter sequence, such as s_1 and s_2.

We have conducted many experiments to evaluate the relationship among the steady-state power, the total power, and the length of the simulation vectors. Figure 1 shows the



(a) 100 samples (each has 2 vectors) (b) 100 samples (each has 1000 vectors)

Fig. 1: The probability density function of power dissipation with respect to sample input sequences with similar steady-state transition power.

probability density function of the total power of a 16-bit ripple-carry adder with respect to two sets of 100 sample input sequences. In these figures, each sample sequence produces the same steady-state power. The sizes of each input sequence in Fig. 1(a) and Fig. 1(b) are 2 and 1000, respectively. For the generation of these two sets of input sequences, we first randomly generate a large number (>> 100) of input sequences with sizes 2 and 1000, respectively. Then each input sequence is simulated to derive the steady-state power. After that, for each size, we randomly select 100 input sequences whose steady-state power is very close to a pre-specified value (the difference is less than 0.1%).

Since the steady-state power for these input sequences are almost the same, *the variation between the samples' power dissipations is mainly due to the hazardous power.* It can be seen from Figure 1 that the standard deviation is dramatically decreased as we increase the *size* of each sample

input sequence. Note that this property is only valid based on the assumption that sample input sequences have very similar steady-state power. Our experimental results are consistent for every module and steady-state power we have tried.

This experiment demonstrates one *simple* but *essential* property. That is, for a given module, input sequences with similar steady-state power will have similar hazardous power when the length of each input sequence is large enough. By definition, these input sequences have similar steady-state power, and therefore, they will have similar total power. For each small range of the steady-state power, we need to find a long enough input sequence that results in the corresponding steady-state power. We can then use a transistor-level or circuit-level power simulator to simulate the input sequence. The derived power can then be used as the expected mean value of the total power for the corresponding steady-state power. The obtained expected mean value can be used to *predict* the total power of the other input sequences that exhibit similar steady-state power.

4. Generating Power Model

From our experiments we observe that, for *fixed* steady-state power, the total power will converge to a *fixed value* when we gradually increase the length of the sample input sequence. Based on this property, we define a one-to-one mapping function between the total power and the steady-state power for each module. This function is called *cross-coefficient function* hereafter. In our approach, the power modeling is referred to as the process that derives this cross-coefficient function. This process is performed only once for each RTL module, and the resulting cross-coefficient function can be used repeatedly for power estimation for any RTL design containing the module. Since the cross-coefficient function is a continuous function, we propose an algorithm to approximate it as a piece-wise linear function.

The main flow for generating the cross-coefficient function for each module is shown in Figure 2. In this algorithm,

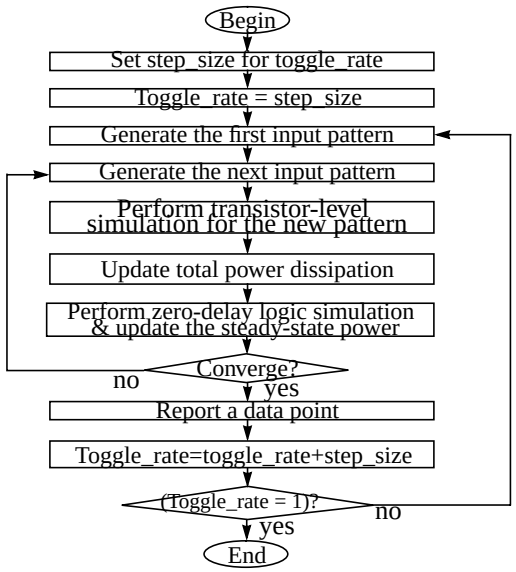


Fig. 2: Power modeling process using a transistor-level power simulator as well as a zero-delay logic simulator.

we set a variable *toggle_rate* for the primary inputs. We vary the toggle_rate to generate a number of sample input sequences with different steady-state power. The toggle_rate is initially set to zero. First, we generate a two-vector input sequence based on the toggle_rate. Next, we gradually increase the length of the sample input sequence based the same value of the toggle_rate. During the process of generating the sample input sequence, the transistor-level power simulation and zero-delay logic simulation are performed for each sequence to derive the corresponding total power and steady-state power, respectively. The process of generating sample input sequences continues until the values of the total power and steady-state power with respect to the sample input sequences have converged within a satisfactory range, respectively. The total power and steady-state power of the sample input sequence is then reported. This corresponds to a data point of the cross-coefficient function for the applied module. At the next iteration, we increase the toggle_rate by a pre-defined step_size to generate the next data point based on the same procedure for the first data point. The power modeling process completes when the toggle_rate reaches 1. The total number of data points for each module generated by this algorithm would be 1/step_size. In our experiment, we use 0.025 as the step_size. Therefore, 40 data points would be generated to approximate the cross-coefficient function of each module. The cross-coefficient functions of all modules are generated based on the above procedure.

Figure 3 shows the cross-coefficient functions for three RTL library modules: a 16-bit array multiplier, a 16-bit ripple-carry adder, and a 16-bit 4x1 multiplexor. It can be seen from this figure that the total power values of the three modules with respect to the same steady-state power are different, and the differences are primarily due to the differences of hazardous power associated with each module.

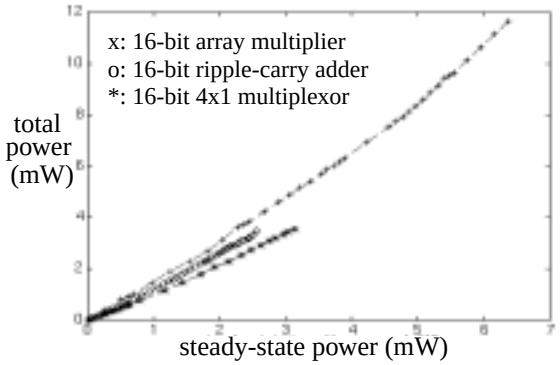


Fig. 3: The comparison of the cross-coefficient functions for three RTL modules.

5. Power Library Interpolation

Most RTL synthesis tools support *parameterized* modules. For instance, a ripple-carry adder can have any number of input bits from 1 to 128. Using the aforementioned power modeling technique for every parameterized module could be very time-consuming. Therefore, we use an interpolation technique to reduce the power modeling time.

Using this power interpolation technique for each type of module, we need to apply the power modeling technique to only a small number of selected modules with representative parameters. For example, for a ripple-carry adder, we may choose only 8-bit, 16-bit, and 32-bit versions for explicit power modeling. The power models of these representative modules are called sample models. For the non-representative modules (e.g., a 10-bit ripple-carry adder), we approximate their power models using interpolation on the sample models, taking the steady-state power as the index. Linear interpolation based on the number of input bits is *not* suitable for some RTL modules, such as multipliers. Therefore, we derive a complexity function for each type of parametrized module. This complexity function relates the structural information (e.g., number of bits, number of input lines, and number of basic cells) to the scale for interpolation. In some sense, the power modeling technique mentioned in the previous sub-section can be regarded as a first-order approximation, while the interpolation using the complexity functions can be regarded as a second-order approximation. We list the complexity functions, denoted as *C*, for a number of modules as follows.

ripple-carry adder and subtracter

$C \propto n_b$, where n_b is the number of bits.

array multiplier

$C \propto n_b(n_b - 1)$, where n_b is the number of bits.

register file

$C \propto \alpha_1 n_b + \alpha_2 n_r$, where n_b and n_r are the number of bits and number of registers, respectively, and α_1 as well as α_2 are constant coefficients.

multiplexor

$C \propto \alpha_3 n_b + \alpha_4 n_i$, where n_b and n_i are the number of bits and number of input lines, respectively, and α_3 as well as α_4 are constant coefficients.

Table 1 shows the estimation errors using this interpolation technique. For the 10-bit modules, we use the power models of the 8-bit and 16-bit modules as the sample models for interpolation. The errors are compared with the results of our power modeling technique described in previous sub-section.

Table 1: Estimation error of power interpolation

10-bit module	ripple-carry adder	array multi-plier	subtracter	4x1 Mux	register file
Error (%)	8.5	10.6	6.7	4.8	6.1

6. Hierarchical Power Estimation Based on Hybrid Power Models

In this section, we present a simulation-based hierarchical power estimation method based on the proposed hybrid power models. The method can handle complex RTL designs consisting of a number of library modules (i.e., power models have been established), and produce the reasonably accurate estimates for the power dissipation of the designs. Given an RTL design, cycle-based RTL simulation is performed to derive the corresponding input sequence for each RTL module, and then the input sequence of each module is used for zero-delay logic simulation to derive the steady-state power of each module. After that, using the pre-computed power models and the information of the steady-state transition power of each module, the *accurate* total power of each RTL module can then be efficiently obtained. This process is extremely efficient because it only requires the zero-delay RTL and logic simulation.

For controllers and random logic circuitry embedded in the RTL design, we apply different approaches to estimate their power dissipation. If the number of primary inputs of a sub-module is small (such as a flip-flop), then we construct a complete power lookup table to replace the cross-coefficient function for the power modeling process. This table is indexed by each possible input vector pair. On the other hand, if the number of inputs of a module is too large (> 4) to be characterized by a complete table, then we adopt the mixed-level Monte-Carlo simulation approach proposed in [6] to compute the power dissipation of the module. Finally, we sum up the power dissipation values of all modules to obtain the total power dissipation of the entire circuit. The flow of the RTL power estimation is shown in Figure 4.

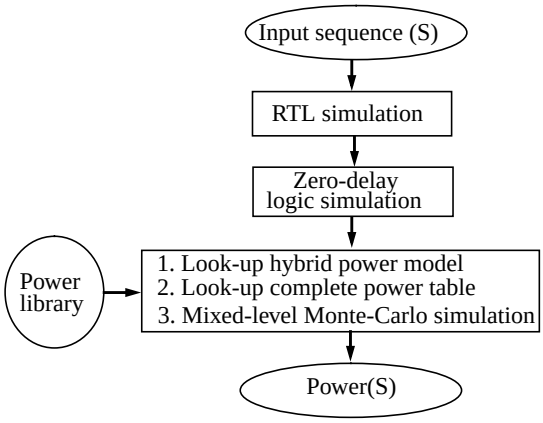


Fig. 4: The flow of our simulation-based RTL power estimation approach.

The possible errors of the estimation based on this power estimation procedure can be classified into two categories: (1) modeling error, and (2) boundary error. The modeling error arises from the statistical inaccuracy of the hybrid power model for each RTL module. The boundary error is due to the glitches power at the interconnects between RTL modules. During the power modeling process, we assume that the arrival times of all inputs for each RTL module are the same. Therefore, the glitches at module boundaries are not considered properly.

7. Experimental Results

We have implemented a prototype tool based on the proposed method. Our tool incorporates the RTL simulator QuickVHDL [10], a zero-delay logic simulator, and a transistor-level power simulator PowerMill [3]. To generate RTL test cases, we use a high-level synthesis system HYPER [2]

to synthesize a number of behavior-level designs into RTL designs. For RTL modules, we synthesis each module, and layout them by a physical design system GARDS [4] using a 0.55μm CMOS library. PowerMill uses the physical design information of each module to achieve circuit-level accuracy for power estimation. The experimental results are obtained by assuming that the clock rate is 10 M Hz. Table 2 shows the modules used in four RTL designs: *ellip*, *wavelet*, and *volterra*.

Table 2: The modules for each RTL design

	ripple-carry adder		sub-tracter		array multi-plier		4-1 MUX		regis-ter file		con.
	# of bits		# of bits		# of bits		# of bits		# of bits		-
	8	16	8	16	16	32	8	16	8	16	-
ellip	0	2	0	0	0	1	0	5	0	5	1
wavelet	0	2	0	1	0	1	0	3	0	7	1
volterra	1	0	0	0	2	0	4	0	6	0	1

Estimation errors for each module

Table 3 shows that the estimation errors of each module based on the hybrid power model. To verify the accuracy of the hybrid power model, we use five input sequences generated by assigning random operands to the inputs of the entire RTL design, and each contains 10,000 patterns. The corresponding input sequences for each module are derived through the cycle-based RTL simulation for the entire RTL design. It should be pointed out that, because of the interaction between the RTL modules, the sequence of each module is highly correlated (temporally and spatially). Columns 2-3 and 5-6 show the results of our approach and transistor-level power simulation using PowerMill for an 8-bit and 16-bit ripple-carry adder. The estimation errors shown in Columns 4 and 7 are compared to the transistor-level simulation (PowerMill) results. The estimation results for other modules are shown in this table. The average errors of our approach is 3.5% compared to PowerMill simulation results.

Estimation errors for each RTL design

Tables 4, 5 and 6 show the power estimation results for four RTL designs: *ellip*, *wavelet*, and *volterra*, respectively. The circuit *ellip* is a 5th order elliptic wave filter, the *wavelet* is a 14th order FIR filter, and the *volterra* is a volterra filter. We apply five input sequences generated in a way mentioned earlier. In each table, Columns 2, 4 and 6 show the results of (1) the steady-state power, (2) our estimation, and (3) running PowerMill for the entire design through the entire input sequence. Columns 3 and 5 show the estimation errors of considering only steady-state power and our approach, respectively. The errors are compared to the results of PowerMill. The CPU times for three approaches are also reported in Columns 7, 8 and 9, respectively.

The estimation errors of our approach are less than 6% for all cases, and the average error is only 4.1%. On the other hand, the average error of only considering the steady-state power could be as high as 44%. The power estimation in our approach is as fast as logic-level approaches using the zero-delay model because it requires only the cycle-based simulation at the RT-level and the logic-level. Also, our approach is about 275 times faster as compared with running PowerMill through the entire input sequence.

8. Conclusion

To achieve more accurate power estimation at the RT-level, we propose a new approach to modeling the power dissipation for RTL modules. Because transistor-level simulation is used for library module characterization, the resulting model is very accurate. Based on the new hybrid power model, the power dissipation of the RTL design can be efficiently estimated by only running RTL simulation and module-by-module logic simulation using the zero-delay model for the given input sequence. The experimental results show that our power estimation time is on average 275 times faster, and the error percentage is less than 6% as compared with the results of running transistor-level simulation for the entire design through the entire input sequence.

Acknowledgement

We thank Dr. David Ihsin Cheng of Exemplar Logic, Inc. for revising the paper and giving many valuable comments.

References

[1] A. Bogliolo, L. Benini, and B. Ricco, “Power Estimation of Cell-Based CMOS Circuits,” *Proc. of Design Automation Conference*, pp. 433-438, June 1996.

[2] A. P. Chandrakasan, M. Potkonjak, J. Rabaey, and R. W. Brodersen, “HYPER-LP: A System for Power Minimization Using Architectural Transformations,” *Proc. Int’l Conf. on CAD*, pp. 300-303, November 1992.

[3] EPIC Design Technology, PowerMill Reference Manual, August 1992.

[4] GARDS, “Command Reference Manual,” Volume 1-4, Silicon Valley Research, September 1996.

[5] C.-T. Hsieh, Q. Wu, C.-S. Ding, and M. Pedram, “Statistical Sampling and Regression Analysis for RT-level Power Evaluation,” *Proc. Int’l Conf. on CAD*, pp. 583-588, November 1996.

[6] S.-Y. Huang, K.-T. Cheng, K.-C. Cheng, and T.-C. Lee, “A Novel Methodology for Transistor-Level Power Estimation,” *Proc. Int’l Symposium on Lower Power Electronics and Designs*, pp. 67-72, August 1996.

[7] P. E. Landman and J. M. Rabaey, “Architectural Power Analysis: The Dual Bit Type Method,” *IEEE Trans. on VLSI Systems*, Vol 3-2, pp. 173-187, June 1995.

[8] J.-Y. Lin, T.-C. Liu, and W.-Z. Shen, “A Cell-Based Power Estimation in CMOS Combinational Circuits,” *Proc. Int’l Conf. on CAD*, pp. 702-707, June 1996.

[9] H. Mehta, R. M. Owens, and M. J. Irwin, “Energy Characterization based on Clustering,” *Proc. of Design Automation Conference*, pp. 702-707, June 1996.

[10] Quick VHDL, “User’s and Reference Manual,” Software Version 8.4_1, Mentor Graphics, July 1994.

[11] C.-Y. Tsui, R. Marculescu, D. Marculescu, and M. Pedram, “Improving the Efficiency of Power Simulators by Input Vector Compaction,” *Proc. of Design Automation Conference*, pp. 165-168, June 1996.

Table 3: Estimation errors for each module

input sequences	ripple-carry adder						subtracter						register file		
	8-bit			16-bit			8-bit			16-bit			8-bit		
	ours (mW)	pow-ermill (mW)	error (%)	ours (mW)	pow-ermill (mW)	error (%)	ours (mW)	pow-ermill (mW)	error (%)	ours (mW)	pow-ermill (mW)	error (%)	ours (mW)	pow-ermill (mW)	error (%)
sequence 1	0.75	0.77	2.6	2.70	2.67	1.1	1.26	1.23	2.4	2.10	2.16	2.8	1.07	1.10	2.7
sequence 2	0.78	0.81	3.7	3.72	3.64	2.2	1.42	1.48	4.1	2.08	2.16	3.7	1.49	1.51	1.3
sequence 3	0.48	0.46	4.3	2.37	2.44	2.9	0.66	0.64	3.1	2.27	2.19	3.7	1.09	1.13	3.5
sequence 4	0.54	0.55	1.8	2.59	2.52	2.8	1.22	1.23	0.8	2.27	2.15	5.6	1.41	1.34	5.2
sequence 5	0.49	0.47	4.3	2.40	2.48	3.2	1.01	0.98	3.1	2.24	2.17	3.2	1.01	0.98	3.1
average	-	-	3.3	-	-	2.4	-	-	2.7	-	-	3.8	-	-	3.2
input sequences	array multiplier						4-1 multiplexor						register file		
	16-bit			32-bit			8-bit			16-bit			16-bit		
	ours (mW)	pow-ermill (mW)	error (%)	ours (mW)	pow-ermill (mW)	error (%)	ours (mW)	pow-ermill (mW)	error (%)	ours (mW)	pow-ermill (mW)	error (%)	ours (mW)	pow-ermill (mW)	error (%)
sequence 1	7.80	7.45	4.7	18.9	19.5	3.1	0.47	0.45	4.4	1.54	1.50	2.7	2.20	2.15	2.3
sequence 2	6.13	5.92	3.5	18.3	19.0	3.7	0.63	0.60	5.0	1.75	1.79	2.2	2.51	2.63	4.6
sequence 3	4.38	4.46	1.8	19.2	18.5	3.8	0.38	0.40	5.0	1.71	1.63	4.9	2.05	1.98	3.5
sequence 4	3.20	3.36	4.8	19.8	18.6	6.5	0.98	0.96	2.1	2.21	2.27	2.6	2.41	2.56	5.9
sequence 5	5.84	6.01	2.8	20.1	19.0	5.8	0.77	0.74	4.1	1.47	1.39	5.8	2.40	2.31	3.9
average	-	-	3.5	-	-	4.6	-	-	4.1	-	-	3.6	-	-	4.0

Table 4: Power estimation of ellip

input sequences	power dissipation (mW)					CPU time (min.)		
	steady-state power		ours		powermill	steady-state power	ours	powermill
	estimation	error(%)	estimation	error(%)				
sequence 1	25.2	37.3	42.1	4.7	40.2	0.30	0.30	132.51
sequence 2	24.9	38.5	38.5	4.9	40.5	0.31	0.31	134.60
sequence 3	22.7	39.3	39.0	4.3	37.4	0.33	0.33	127.96
sequence 4	23.1	40.3	40.1	3.6	38.7	0.31	0.31	138.53
sequence 5	24.1	36.7	36.3	4.7	38.1	0.29	0.29	131.24
average	-	38.4	-	4.4	-	0.31	0.31	132.97

Table 5: Power estimation of wavelet

input sequences	power dissipation (mW)					CPU time (min.)		
	steady-state power		ours		powermill	steady-state power	ours	powermill
	estimation	error(%)	estimation	error(%)				
sequence 1	30.0	48.1	60.6	4.8	57.8	1.12	1.12	290.76
sequence 2	28.1	50.2	58.2	3.2	56.4	1.05	1.05	292.58
sequence 3	29.2	47.3	53.7	3.1	55.4	1.08	1.08	289.13
sequence 4	30.8	46.1	60.1	5.3	57.1	1.12	1.12	287.54
sequence 5	29.0	48.9	54.7	3.7	56.8	1.10	1.10	290.43
average	-	48.1	-	4.0	-	1.09	1.09	290.09

Table 6: Power estimation of volterra

input sequences	power dissipation (mW)					CPU time (min.)		
	steady-state power		ours		powermill	steady-state power	ours	powermill
	estimation	error(%)	estimation	error(%)				
sequence 1	13.9	45.1	24.0	5.1	25.3	0.74	0.74	91.27
sequence 2	13.9	43.3	25.7	4.9	24.5	0.73	0.73	94.53
sequence 3	14.5	44.0	26.4	1.9	25.9	0.69	0.69	90.18
sequence 4	13.9	45.9	26.7	3.9	25.7	0.71	0.71	92.54
sequence 5	13.5	46.2	24.3	3.2	25.1	0.71	0.71	91.08
average	-	44.9	-	3.8	-	0.72	0.72	91.92