Probabilistic Bottom-up RTL Power Estimation

Ricardo Ferreira A-M.Trullemans
DICE - Université Catholique de Louvain
Place Levant 3, 1348 Belgium

{cacau,anmarie}@dice.ucl.ac.be

José Costa José Monteiro
IST/INESC
Rua Alves Redol 9, 1000 Lisboa - Portugal

{jccc,jcm}@inesc.pt

Abstract

We address the problem of power estimation at the register-transfer level (RTL). At this level, the circuit is described in terms of a set of interconnected memory elements and combinational modules of different degrees of complexity. We propose a bottom-up approach to create a simplified high-level model of the block behavior for power estimation, which is described by a symbolic local polynomial. We use an efficient gate-level modeling based on the *Polynomial Simulation* method and ZBDDs. We present a set of experimental results that show a large improvement on performance and robustness when compared to previous approaches.

1 Introduction

Power optimization of VLSI circuits can be accomplished at the various levels of design abstraction. Higher level approaches can exploit a larger degree of freedom, therefore can have a larger impact in the reduction of the power consumption. However, the power estimation can be inaccurate at this level. In contrast, power consumption can be accurately estimated at lower level approaches, but the degrees of freedom are more restricted in terms of providing a significant optimization. A promising approach [2, 6] is to use tools used at lower levels of abstraction to characterize higher level modules.

A circuit described at the register-transfer level (RTL) consists of a set of interconnected functional units, such as arithmetic, logic, memory and registers. A simple RTL description can represent a complex gate level circuit. It contains enough information to describe the circuit behavior, and then different techniques may be applied to transform and optimize the original RTL description. However, we need more information to estimate the power dissipation of different RTL design alternatives. The functional units come from a well characterized library. A good RTL power estimation method should build an accurate model of each unit by extracting information from the library, such as gate and connection capacitances. Moreover, the model should be simple enough to quickly calculate the total unit power dissipation. The characterization step builds a simplified high-level model by using the lower level details. This step is executed only once, and may be time consuming. However, the evaluation step, which is repeatedly called during the RTL synthesis to compute the power of the functional unit, should be simple and fast.

In this work, we propose a bottom-up approach based on ZBDDs that uses the gate-level *polynomial simulation* (PS) method [7, 8] to estimate the power of RTL circuits. Our approach extracts parameters from the lower design abstraction level to build an efficient and accurate model at high level. Our approach is probabilistic, and therefore does not suffer from input pattern dependence. It also offers a trade-off accuracy/performance parameter, and takes into account glitch power.

This paper is organized as follows. Section 2 presents a short review of several types of decision diagrams, namely ZBDDs which we use as an efficient representation for the polynomials needed by the PS method. Section 3 addresses the previous work for RTL power estimation. Section 4 reviews the main features of the PS method at gate-level. Section 5 presents our PS method based on ZBDDs for RTL circuits. Finally, the experimental results show a large improvement on performance without losing in accuracy with respect to the exact symbolic method [16].

2 Decision Diagrams

Boolean functions [12] may be used to represent a set of combinations. In a Boolean space of n variables, a n-bit vector is used to represent each combination. If the variable x_i belongs to the combination, the bit i is set to 1, otherwise it is set to 0. A set of combinations could be represented by the ON-set, which is composed of the set of n-bit vectors. For example, the set $s = \{\{x_1\}, \{x_2\}\}$ over the Boolean space $\{x_1, x_2, x_3\}$ will be represented by the ON-set $((100), (010))$ or the Boolean function $f = x_1\ \overline{x_2}\ \overline{x_3} + \overline{x_1}\ x_2\ \overline{x_3}$. Using BDDs for the Boolean functions, we can manipulate sets of combinations efficiently. However, the

BDD size depends on the number of input variables in the Boolean space, as shown in Fig. 1a, where the set of combinations $s = \{\{x_1\}, \{x_2\}\}$ is defined over a Boolean space of 3 and 4 variables, respectively. Even if a variable x is not present in a combination, the corresponding 1-path has a node with x whose 1-edge points to the 0-Terminal node. The ZBDD was proposed in [15] to overcome this drawback. The basic reduction rule is to eliminate nodes whose 1-edge points to the 0-Terminal node. The node decomposition $f = vf_1 + f_0$ is called *linear decomposition*. If $f_1 = 0$ the reduction rule is applied, so $f = v \cdot 0 + f_0 = f_0$. The set of combinational representation is independent of Boolean space size, as shown in Fig 1b. Therefore, ZBDDs are more compact than BDDs in the presence of sparse sets. BDDs are still very efficient to represent Boolean function for general applications, while ZBDD are more efficient for representation of set of combinations. Another BDD variants have been proposed for specific applications, for instance *Algebraic Decision Diagrams* (ADDs) [1], which are derived from the BDDs by adding multiple terminal nodes (Fig. 1c). In addition to the Boolean operations, ADDs can handle arithmetic operations, such as addition, and find minimal and maximal terms. ADDs are based on Shannon decomposition (like BDDs), where the assignment $v = 1$ is encoded as v, and the assignment $v = 0$ is encoded as $(1 - v)$. The ADD node decomposition is $f = v \times f_v + (1 - v) \times (f_{\overline{v}})$. Other variant is the Binary Moment Diagram (BMD)[22], which is an efficient representation for formal verification of arithmetics circuits. BMDs have multi-terminal nodes like ADDs, but use the *linear decomposition* like ZBDD.

3 Related Work

There are two main approaches to characterize the functional units: statistical (also known as dynamic or simulation-based approaches) and probabilistic (or static approaches).

Statistical approaches [11, 19, 21] simulate a given unit repeatedly using a large set of input vectors, and obtain a simple function to represent the power dissipation. The functions have typically few parameters, namely an effective capacitance for each unit, a scaling factor in terms of the word size [11, 19], and a temporal correlation coefficient that represents the correlation between two consecutive word level values. The major drawback of statistical techniques is the input pattern dependence, which limits their practical application because the estimation corresponds directly to the input patterns that were used to drive the simulation. Therefore, the characterization step may not correspond to the context of the real application. Some recent work have been proposed to partially overcome the input pattern dependence of statistical approaches [3, 20].

On the other hand, probabilistic approaches are input pattern independent, and use a compact representation for the input conditions. The entropy concept [13, 17] was introduced to compactly represent the average switching activity and characterize the RTL unit.

Recently, Bogliolo [2] proposed a simplified model based on ADDs to represent the circuit. A characterization step builds an ADD, which is an implicit and compact representation of the whole circuit for all possible input vectors, to compute the total power under the zero delay model. However, the ADD size grows up exponential as a function of the number of primary inputs. An on-the-fly heuristic is proposed to limit the maximal size, where the size is checked to see if its exceeds the maximum allowed value while the ADD is being built. The heuristic collapses the nodes with minimum variances. No assumption is made about the input vector sequence during the characterization step. Still, glitch power and circuit topology are not taken into account.

More recently, Costa *et al.* [6] propose a probabilistic approach which takes into account the glitch power. This approach can handle different input statistics and builds a bottom-up model from the gate-level analysis. The approach is based on *polynomial simulation* PS [7] to compute the power estimation at gate-level. The PS method substitutes some variables by their probability values to reduce the size of the probability polynomials. However, this reduction can not be applied in RTL estimations, because we need to express the polynomial in terms of the primary input variables. Costa *et al.* [6] propose another heuristic based on the least squares method [10]. Each input x has four probability variables: $p^{00}(x), p^{01}(x), p^{10}(x)$, and $p^{11}(x)$, corresponding to the input staying low $(0 \rightarrow 0)$, making a rising transition $(0 \rightarrow 1)$, a falling transition $(1 \rightarrow 0)$, and staying high $(1 \rightarrow 1)$. The RTL unit is evaluated for different combinations of the probability values, and the least squares method [10] is used to reduce the polynomial size. The approximate polynomial is as follows:

$$P = a_0 + \sum_{i=1}^{n} [a_{3i-2} p^{00}(x_i) + a_{3i-1} p^{01}(x_i) + a_{3i} p^{10}(x_i)]$$

where the terms a_j's are computed using the least squares method. The characterization step uses a sequence of input probabilities to build the final polynomial. The polynomial with n inputs is evaluated at several points in the input space. However, if the input probabilities under the operating conditions

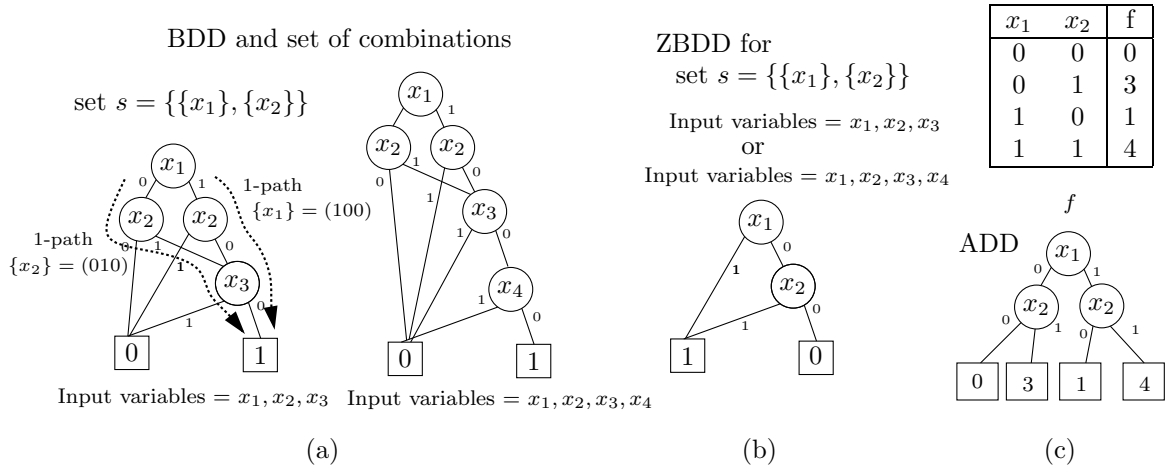

Figure 1: (a) BDD for a set of combinations (b) ZBDD (c) ADD for a multi-valued function

are far from these points used in the characterization step, the accuracy may be reduced.

In this paper, we propose a different heuristic to reduce the size of the polynomials, based on ZBDDs [14]. Our approach is independent on the input probability: all input variables are handled by a symbolic representation during the characterization step, and no constants are introduced, in contrast to the least squares approach [6]. Section 5 details our approach.

4 PS method

The *polynomial simulation* PS method [7] is a generalization of the exact signal probability evaluation method due to Parker and McCluskey, which as been extended to handle temporal correlation and arbitrary transport delays. The PS method is parameterized by a single parameter l, determining the speed-accuracy tradeoff. This allows the user to improve the accuracy at the expense of runtime. In propagating signal probabilities through a logic circuit, spatial correlation measures how the probabilities of the inputs to a gate are related. The PS method uses the *suppression rule* introduced by Parker-McCluster [18], which consists of suppressing all high order exponents (≥ 2), to handle the spatial correlation. In addition, the PS method handles the first order temporal correlation thanks to the Cheng's *removal rule* [4]. All products $x^{ij} \cdot x^{kl}$, where $i \neq k$ and/or $j \neq l$, are removed because the signal x can not have two different transitions at the same time.

In the presence of reconvergent paths, for an exact computation of switching activity, each local polynomial of an internal node has to be expressed as a function of the primary input nodes, and this can be very time-consuming. The PS method proposes to use a limited depth computation to find a good compromise between speed and accuracy. To reduce the complexity of the polynomials some variables may be replaced by their probability values. The reason why the substitution can be applied is because the variables outside the reconvergent path will not generate exponential terms. As we have seen, spatial correlation between signals appears as exponential terms that have to be considered. The constant substitution is applied only to the variables that are not in a reconvergent path or if reconvergent paths meet after l logic levels. The variables which reconverge before l levels are referred as the *active variables*. Even if the paths meet after l levels, the effect of exponential terms is negligible as compared to the local node probability. The rationale behind this approximation scheme is that spatial correlation between internal signals is more important when reconvergent paths meet within a few logic levels. Recently, [8] proposes an efficient implementation of the PS method thanks to practical issues for computing local node activity: the computation is free of internal correlation (no temporary active variables are needed), and the polynomial representation is based on Probability BMD, that are derived from BMD by handling the suppression and reduction rules [8]. The constants are represented by the multi-terminal nodes.

5 Local Symbolic Polynomial

The Probability BMD can improve the performance of the PS method in presence of a large set of active variables as show the experiments done in [8]. More-

over, the local computation free of internal correlation replaces the temporary active variable by constants. As the polynomial should be expressed in terms of primary input variables at RT-level, we propose to use the ZBDDs, because the constants are not present. Our approach is called Local Symbolic Polynomial method, LSZBDD method for short.

5.1 Characterization

We first compute the active region with a limited depth. The *feather* algorithm [5] is used to find the local minimum input support-set of the active region. The root node is expanded until reaching the local inputs. All local inputs are preserved without substituting any variable inside the active region, in contrast to the PS method which substitutes the local variables outside of the reconvergent paths. The trade-off accuracy/performance is controlled by the depth used to compute the active region.

In our no-substitution approach, the characterization could be time-consuming, and builds the symbolic local ZBDDs for all internal nodes. However during the evaluation step, the local ZBDDs have only to be traversed to recompute the switching activity, which is very fast. The total power is computed as follow.

$$power = \sum_{\forall g_i} C_{g_i} \cdot \sum_{\forall t} (g_i^{01}[t] + g_i^{10}[t]) \qquad (1)$$

where $g_i^{01}[t]$ and $g_i^{10}[t]$ represent the switching activity at transition time t, and C_{g_i} the capacitance load driven at the output of gate g_i. Given an input probability vector, the circuit is traversed from the primary inputs to the primary outputs, and the ZBDD polynomials are evaluated.

Example 1 *Let us consider a simple functional unit shown in Fig 2a. Suppose the unit delay model. The transition time set for gate g_1 is $\{1\}$, and for g_2 is $\{1, 2\}$. There are no reconvergent paths, so the gate polynomials are expressed in terms of local inputs. The characterization step computes all local symbolic polynomials. The Fig 2b shows the ZBDD polynomial corresponding to the transition $0 \rightarrow 1$ for the gate g_1 at time $t = 1$, which is a function of primary input variables. The polynomial for the gate g_2 has g_1 as a local symbolic variable, as shown in Fig 2c.*

Our approach, as seen in the previous example, uses hierarchical ZBDD variables, which reduces the ZBDD sizes. This approach is equivalent to a functional decomposition of the polynomials by introducing intermediate variables. The Boolean space consists on input and intermediate variables, and it can be very large. However, the local polynomials are expressed in terms of few variables (sparse sets), so the ZBDD is a very compact representation thanks to the *linear decomposition*. An internal reconvergent path introduces some spatial correlations, which needs to be taken into account to preserve the accuracy. As mentioned before, our reduction criterion is based on the analysis of these paths, that are extracted from the circuit internal structure. In contrast to the variance used to reduce the ADD in [2], which is based on the Boolean function. ADDs (like BDDs) are based on Shannon decomposition, and are not efficient for sparse sets.

We preserve the PS method features, where the capacitances and the delay are input parameters to the evaluating process. The library model could give an accurate measure of the internal capacitances, which could take into account the gate and the connection capacitances extracted after the place and route step. At this level, the delay will be also more accurate.

5.2 Estimation

In a first step, all functional units are characterized, and the ZBDD polynomial for each unit is stored in a table. We restrict the discussion here to the characterization of combinational modules, however the evaluation process could be used to estimate the power of a sequential or pipeline circuit [23, 6]. Given the input probability values, the RTL circuit is traversed in depth first order. The probability polynomial at the output of each functional unit is computed and applied to the input of the next module(s). Normally, a functional unit has many transition instants at the output. Suppose that the inputs of unit C are connected to the outputs of units A and B. Suppose that A has output transitions at instants t_1, t_2 and B at instant t_3. We apply the same approach, which was introduced at the gate-level for the PS method [7], to the unit-level. The module C is evaluated three times, one for each input transition instant. The total power is computed by summing all the partial power terms over each evaluation.

6 Experimental Results

We have analyzed some typical examples with this technique, as experiments. We compare our LSZBDD heuristic and the least squares approximation [6].

The first set of results lists the accuracy and performance for the characterization step over a set of 8-bit functional units extracted from [6]. Table 1 lists the results obtained for both heuristics under the unit

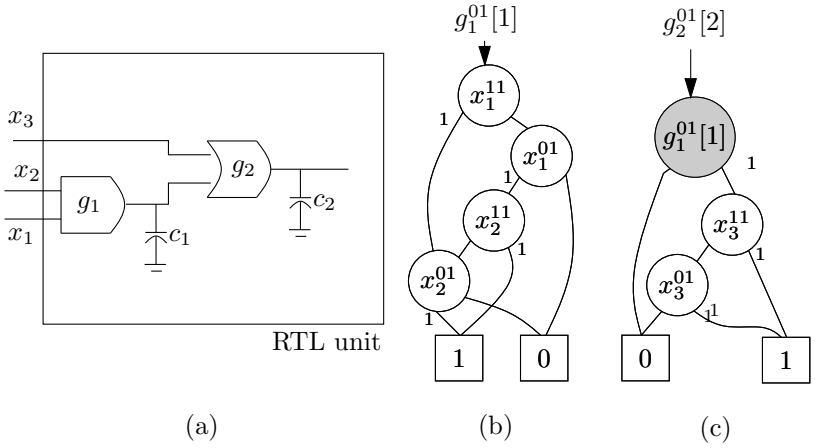

Figure 2: (a) A gate-level description of functional unit (b) The ZBDD for the polynomial $g_1^{01}[1]$ (c) The ZBDD for the polynomial $g_2^{01}[2]$

Module Name	Input Prob	Least Square char time	Least Square Error avg	Least Square Error max	Local Symbolic ZBDD time char	time eval	Error avg	Error max	Size Nodes
add_cla8	A	87.3m	.3	.3	.8	.1	.23	.23	2423
	B		.83	.9			.28	.39	
	C		8.1	8.1			.23	.61	
add_rpl8	A	72.2m	.4	.4	.5	.1	.22	.22	1451
	B		.33	.8			.26	.32	
	C		7.4	7.4			.20	.29	
add_cbp8	A	75.0m	.7	.7	.5	.1	2.5	2.5	1724
	B		.73	.9			1.95	3.8	
	C		11.3	11.3			2.1	4.2	
sub8	A	4.6h	.7	.7	1.2	.15	.5	.5	5819
	B		1.1	1.7			.67	1.25	
	C		.4	.4			1.0	2.0	
barrel_shifter4_8	A	22.3s	.0	.0	1.6	.2	5.1	5.1	11016
	B		1.5	4.0			4.4	6.6	
	C		11.7	11.7			3.3	4.3	
equal8	A	9.6m	.0	.0	.15	.02	.0	.0	330
	B		.07	.1			.0	.0	
	C		6.3	6.3			.0	.0	
greater8	A	40.7m	1.3	1.3	.5	.04	2.2	2.2	1665
	B		.7	1.2			1.5	2.2	
	C		1.5	1.5			1.3	2.6	
mux8	A	.2s	.0	.0	.2	.1	.0	.0	470
	B		.0	.0			.0	.0	
	C		.0	.0			.0	.0	

Table 1: Power Estimation for functional Unit Characterization

delay model. The first column lists the functional unit names: carry-lookahead (add_cla), ripple-carry (add_rpl) and a carry-bypass (add_cbp) adders, one subtractor (sub), greater-than (greater) and equality (equal8) comparators, one multiplexor (mux) and a 4-bit barrel-shifter (barrel_shifter). The column 2 lists the input probability type:

A the probabilities are equal to .25 for all inputs;

B the probabilities ranges randomly from .1 to .25;

C the probabilities ranges randomly from .0 to .25;

Column 3,4 and 5 are extracted from [6], and show results obtained by the least squares heuristic: the characterization time under an Ultra Sparc I with 384Mb, the average and maximal error for the total power, respectively. The total power computed by the exact symbolic method is used as reference [9, 16]. The maximal and average error are identical for the input probability type A, because there is only one

possibility. The type B was computed for three different input vectors and the type C was computed just once in [6]. Column 6,7,8,9 and 10 list the results obtained by the LSZBDD heuristic. The average and maximal errors (columns 8 and 9) was computed over 10 different input probabilities. The CPU time is in seconds and corresponds to a computation on an Ultra Sparc II with 128Mb. Column 10 lists the ZBDD size. The maximal local support is set to 4 during the LSZBDD computation.

We observe that the characterization step is time expensive (more than 4 hours for the subtractor) for the least squares heuristic. The maximal characterization time obtained by our LSZBDD heuristic is below 2 seconds. However, the characterization time is computed just once, and could be high. The least square heuristic is calibrated by using input probability close to the types A and B. Therefore, the errors are relatively small for these input probabilities, however when the input probability type C is used, the error could be around 10% for many circuits. Our LSZBDD heuristic has a maximal error under 6.6%. The error is not sensitive to the input probability type. The error is due to the internal correlation generated by the reconvergent paths. As the PS method, our LSZBDD is parameterized and could be more accurate when taking more levels into account during the active variable selection. The barrel shifter is the worst example, where each output is a function of all primary inputs, and a high correlation degree is present. However, the LSZBDD heuristic gives better results for most of the circuits at different input probability types: the multiplexor and the equality comparator are exactly computed, and the carry lookahead and the ripple carry adders are very close to the exact solution ($< .6\%$).

Table 2 shows the RTL circuit described in terms of the functional units, which are used to evaluate the least squares method in [6]. Table 3 lists the results obtained by least squares heuristic and our LSZBDD heuristic to compute the total power of RTL circuits. Column 1 lists the circuit names. Columns 2 and 3 list the total power and the cpu time obtained by the exact symbolic method [16], respectively. Columns 4 and 5 list the error with respect to the power dissipation computed by the exact method, and the cpu time in seconds for the least squares heuristic. Finally, columns 6 and 7 list the results obtained by our LSZBDD heuristic.

The CPU time for both heuristics are similar, and a large speedup factor is obtained (up to orders of magnitude) with respect to the exact symbolic method. Moreover, our LSZBDD heuristic significantly reduces the error below 2%, which is due to the independence of input probability values during the characterization step.

Table 4 lists some 16 bit adders: carry-lookahead (add_cla), ripple-carry (add_rpl) and a carry-bypass (add_cbp). The characterization is done by limiting the local support to 4 and 8 variables. The column description is similar to table 1. We observe a significantly improvement on the accuracy when 8 variables are taken into account to compute the local support. The characterization and the evaluation time are still small.

7 Conclusions and Future Work

We have shown a novel method for power estimation at RT-Level. This method employs gate-level polynomial simulation to accurately model RTL modules. The use of ZBDDs allows for effective and simple high level models.

The input probabilities are kept symbolic at the characterization step in order to avoid wrong assumptions about the circuit operating conditions. The experimental results have shown a performance improvement of orders of magnitude with respect to the exact symbolic method [16] without losing accuracy. Our approach take into account glitches. The power dissipation due to glitches is typically around 20% of the total power, however for some cases such as arithmetic circuits, the glitch power could be as high as 70% of the total power [9].

A more complex characterization step could be developed. First, the local support could be computed by modifying the original feather algorithm [5] to take into account a limited depth input set instead of the primary inputs. Then, a characterization step may be performed by using 10 or more variables. However, a large ZBDD will be generated. A post-processing step can filter out the polynomial products with a large number of variables: as the probabilities ranges from 0 to 1, a probability product of 5 variables or more could be ignored. Finally, a compact and approximate ZBDD is used to perform the evaluating step. Future works will be devoted to apply these techniques in high level synthesis tools.

References

[1] Bahar, I., Frohm, E., Gaona, C., Hachtel, G., Macii, E., Pardo, A., and Somenzi, F. Algebraic decision diagrams and their applications. In ACM/IEEE International Conference on Computer-Aided Design (November 1993).

Circuit	add	sub	shift	equal	greater	mux
mux_adder	1	0	0	0	0	2
mux_adder8	2	0	0	0	0	5
mux_adder16	4	0	0	0	0	13
add_if_shift	1	0	1	1	0	1
add_carry_select	3	0	0	0	0	1
add_or_sub	1	1	0	0	0	1
mux_sub	0	2	0	0	1	1
gcd	1	0	0	0	1	2

Table 2: Some RTL circuits

Circuit name	Exact		Least Square		Local Symbolic ZBDD	
	pow	time	% error	time	% error	time
mux_adder	736	3.1	0.1	0.4	0.01	0.2
mux_adder8	1754.5	9.8	5.7	0.5	0.08	0.85
mux_adder16	3851	332.7	9.0	0.7	0.1	2.3
add_if_shift	1384	109.5	11.8	0.6	1.7	0.9
add_carry_select	1460	4.3	11.8	0.6	1.2	0.9
add_or_sub	1304	5.8	1.9	1.1	1.8	0.8
mux_sub	1821	8.8	0.6	1.1	0.5	.9
gcd	1225	44	1.9	0.8	1.3	1.0

Table 3: Power estimation for RTL circuits

Module name	Input prob	Local Symbolic ZBDD				
		time		% error		Size
		char	eval	avg	max	Nodes
local support = 4						
add_cla16	A	2.5	.15	2.0	2.0	10895
	B			1.7	2.7	
	C			1.8	3.2	
add_rpl16	A	3.8	.2	18	18	20213
	B			16.3	20.5	
	C			15.3	19.7	
add_cbp16	A	5.4	.3	15.0	15.0	31912
	B			13.4	16.2	
	C			11.3	18.6	
local support = 8						
add_cla16	A	2.8	.15	.65	.65	10877
	B			.52	1.0	
	C			.58	2.0	
add_rpl16	A	7.0	.2	1.3	1.3	18664
	B			1.1	1.3	
	C			.86	.1.25	
add_cbp16	A	10	.4	5.2	5.2	40044
	B			4.8	6.2	
	C			4.8	5.6	

Table 4: 16 bit adders characterization

[2] Bogliolo, A., Benini, L., and Micheli, G. D. Characterization-free behavioral power modeling. In DATE - Design, automation and test in Europe (February 1998).

[3] Chen, Z., and Roy, K. A power macromodeling technique based on power sensitivity. In ACM/IEEE Design Automation Conference (June 1998).

[4] Cheng, D. Power Estimation of Digital CMOS Circuits and the Application to Logic Synthesis for Low Power. PhD thesis, University of California at Santa Barbara, 1995.

[5] Choi, H., and Hwang, S. Improving the accuracy of support-set finding method for power estimation of combinational circuits. In European Design and Test Conference 97 (March 1997), pp. 526–530.

[6] Costa, J., Monteiro, J., Silveira, L. M., and Devadas, S. A probabilistic approach for rt-level power modeling. In The 6th IEEE International

Conference on Electronics, Circuits and Systems (September 1999).

[7] Costa, J. C., Monteiro, J. C., and Devadas, S. Switching activity estimation using limited depth reconvergent path analysis. In Proceedings of the 1997 international symposium on Low power electronics and design (1997), pp. 184–189.

[8] Ferreira, R., and Trullemans, A.-M. An efficient switching activity estimation for low power resynthesis under general delay. In PATMOS'99 International Workshop (October 1999).

[9] Ghosh, A., Devadas, S., Keutzer, K., and White, J. Estimation of average switching activity in combinational and sequential circuit. In ACM/IEEE Design Automation Conference (june 92), pp. 253–259.

[10] Golub, G. H., and Van Loan, C. F. Matrix Computations, 2nd ed. Johns Hopkins University Press, Baltimore, 1989.

[11] Landman, P., and Rabaey, J. Activity-sensitive architectural power analysis. IEEE Transactions on Computer-Aided Design 15, 6 (June 1996), 571–587.

[12] Malik, S., Wang, A. R., Brayton, R. K., and Sangiovanni-Vincentelli, A. L. Logic Verification using Binary Decision Diagrams in a Logic Synthesis Environment. In ACM/IEEE International Conference on Computer-Aided Design (1988).

[13] Marculescu, D., Marculescu, R., and Pedram, M. Information theoretic measures of energy consumption at register transfer level. In Proceedings 1995 international symposium on Low power design (1995).

[14] Minato, S. "Zero-Suppressed BDDs for Set Manipulation in Combinatorial Problems". In 30th ACM/IEEE Design Automation Conference (1993), pp. 272–277.

[15] Minato, S. Binary Decision Diagrams and Applications for VLSI CAD. Kluwer, Norwell/MA, USA, 1996.

[16] Monteiro, J., Devadas, S., Ghosh, A., Keutzer, K., and White, J. Estimation of average switching activity in combinational logic circuits using symbolic simulation. IEEE Transactions on Computer-Aided Design 16, 1 (1997), 121–127.

[17] Najm, F. N. Towards a high-level power estimation capability. In Proceedings 1995 international symposium on Low power design (1995).

[18] Parker, K., and McCluskey, E. Probabilistic treatment of general combinational networks. IEEE Transactions on Electronic Computers (1975).

[19] Powell, S., and P.Chau. Estimating power dissipation of vlsi signal processing chips: the pfa technique. In VLSI Signal Processinf IV (1990).

[20] Qiu, Q., Wu, Q., Pedram, M., and Ding, C.-S. Cycle-accurate macro-models for rt-level power analysis. In Proceedings 1997 international symposium on Low power design (1997).

[21] Raghunathan, A., Dey, S., and Jha, N. K. Register-transfer level estimation techniques for switching activity and power consumption. In Proceedings of the IEEE/ACM International Conference on Computer-Aided Design (Washington, Nov. 10–14 1996), IEEE Computer Society Press, pp. 158–165.

[22] R.E. Bryant, and Y.-A. Chen. Verification or Arithmetic Circuits with Binary Moment Diagrams. In 32nd ACM/IEEE Design Automation Conference (Pittsburgh, June 1995), Carnegie Mellon University.

[23] Tsui, C.-Y., Monteiro, J., Pedram, M., Devadas, S., Despain, A. M., and Lin, B. Power estimation in sequential logic circuits. IEEE Transactions on VLSI Systems (September 1995).