Transactions Briefs

Least-Square Estimation of Average Power in Digital CMOS Circuits

Ashok K. Murugavel, N. Ranganathan, R. Chandramouli, and Srinath Chavali

***Abstract*—The estimation of average-power dissipation of a circuit through exhaustive simulation is impractical due to the large number of primary inputs and their combinations. In this brief, two algorithms based on least square estimation are proposed for determining the average power dissipation in complementary metal–oxide–semiconductor (CMOS) circuits. Least square estimation converges faster by attempting to minimize the mean square error value during each iteration. Two statistical approaches namely, the sequential least square (SLS) estimation and the recursive least square estimation are investigated. The proposed methods are distribution independent in terms of the input samples, unbiased and point estimation based. Experimental results presented for the MCNC'91 and the ISCAS'89 benchmark circuits show that the least square estimation algorithms converge faster than other statistical techniques such as the Monte Carlo method [4] and the DIPE [8].**

***Index Terms*—Average power estimation, recursive least squares, statistical.**

I. Introduction

THE GROWTH in the use of personal computing, wireless communication, and portable devices, has increased the need for low power, high performance, compute intensive VLSI circuits. Thus, designing efficient techniques for power estimation is important and numerous works have appeared in the literature. The methods for estimation of average power can be classified into two major categories: *i) simulative* and *ii) nonsimulative*. In simulative approaches [4], [11], [12], the circuit is simulated for different input combinations and the average power is calculated as an average of these simulated values. Exhaustive simulation is accurate and takes care of spatial and temporal correlations within the circuit, however, it is time consuming. Nonsimulative approaches can be classified as *i) probabilistic* and *ii) statistical*. In probabilistic techniques [13]–[15], the input probabilities are propagated through the entire circuit and the switching probabilities at each node are computed. Probabilistic techniques are weakly pattern dependent. They are fast and tractable but typically involve assumptions about joint distribution. In macromodeling approaches [16], gate-level circuits are characterized with parameters such as the input-signal probability, the input- and the output-transition densities. Also methods for computing the lower and upper bounds exist [11], [12].

In statistical techniques [4]–[10], a circuit is simulated for a set of input vectors and the outputs are monitored where a stopping criterion is used to determine when to stop the simulation. The first contribution in this area is the Monte Carlo approach [4]. For randomly generated inputs, the power-values output by the simulations are observed and the simulation is continued based on the mean and the standard deviation of the observed values. The desired accuracy is specified using confidence intervals. In [7], statistical simulation is used to estimate the individual node transition density and hence, the average power for the circuit. In [5], [8], a distribution independent power estimator (DIPE) is proposed for computing the switching activity of the circuit. The simulation for switching activity is terminated using a stopping criterion based on the *central limit theorem*. In [9], the power-estimation problem is transformed into a survey-sampling problem and a stratified random sampling approach is used to take care of multimodal distributions within the sample set.

The above statistical methods are confidence interval based and as a result, require more iterations to converge. In this brief, two statistical algorithms based on the least square estimation technique are investigated for average power estimation: *i)* sequential least square (SLS) estimation and *ii)* recursive least square (RLS) estimation.

II. Least Square Estimation

The estimation problem can be viewed as a special case of the approximation problem. Approximation problems can be viewed as finding the approximate value of an unknown variable from a combination of a known set of variables. The primary goal of the least square estimation technique [2] is to find a good estimator, which is unbiased, has minimum variance, and most important, no probabilistic assumption is made about the data.

In least square estimation techniques, the given data, $x[n]$, deviates from the model generated samples $s[n]$ because of the inaccuracies in the model. The closeness of the two-data points is given by

$$J_N = \sum_{n=0}^{N-1} (x[n] - s[n])^2. \tag{1}$$

The least square estimate minimizes the function J_N which is the squared difference of the given data $x[n]$ and the unknown data $s[n]$. The instantaneous error is the amount of deviation of $x[n]$ from $s[n]$. Let $s[n] = A$ for all n, where A denotes the average-power values of a circuit. According to the least squares technique, A can be estimated by minimizing (1) using the simulated power data $x[n]$. Differentiating (1), with respect to A and equating to zero

$$\hat{A} = \frac{1}{N} \sum_{n=0}^{N-1} x[n]. \tag{2}$$

Thus, the square of the error between the deterministic signal $s[n]$ and the observed data $x[n]$ is minimized.

In this confidence interval based estimation technique, the $(1-\alpha)$ confidence level is given as

$$\left(\overline{X} - \left(\frac{s_T}{\sqrt{N}} t_{\alpha/2}\right), \overline{X} + \left(\frac{s_T}{\sqrt{N}} t_{\alpha/2}\right)\right) \tag{3}$$

where $\overline{X}$ is the mean of samples, s_T is the standard deviation, N is the number of observed data values and $t_{\alpha/2}$ is obtained from the t-distribution. Therefore, the length of this interval is

$$\left(\frac{2s_T}{\sqrt{N}}\right) t_{\alpha/2}. \tag{4}$$

The length of the interval may not be a constant. The standard deviation might be large because of inaccuracies in the model. As the length of this interval increases, the error in the estimated average power value

A. K. Murugavel and N. Ranganathan are with the Center for Microelectronics Research, Department of Computer Science and Engineering, University of South Florida, Tampa, FL 33620 USA.

R. Chandramouli is with the Department of Electrical and Computer Engineering, Stevens Institute of Technology, Hoboken, NJ 07030 USA.

S. Chavali is with LSI Logic, Santa Clara, CA USA .

Publisher Item Identifier S 1063-8210(02)00483-3.

increases. This is the main discrepancy in the Monte Carlo technique [4], which is a confidence interval based estimation technique. The interval should be short for the given confidence level $(1-\alpha)$, to accurately estimate the power dissipated.

A. SLS Algorithm

In the application of statistical estimation techniques for computing the average power, as more power values are available as time progresses, we have the option of waiting until all the data is available or processing the observed data sequentially in time. In the SLSss approach [2], the average power is updated sequentially in time as each of the observed power values $x[0],\ x[1],\ \ldots\ x[N-1]$ becomes available.

Consider a diagonal matrix W with diagonal elements $[W]_{ii} = \omega_i > 0$, then, the least squares error equation is

$$J(A) = \sum_{n=0}^{N-1} \omega_n (x[n] - A)^2 \tag{5}$$

where ω_i is a diagonal element of the matrix $[W]$. $J(A)$ is the least square error of the estimate computed using the observed data $x[n]$. Assuming $x[n] = A + \overline{\epsilon}[n]$ where $\overline{\epsilon}[n]$ is the error due to model inaccuracies with variance σ_n^2, then $\omega_n = 1/\sigma_n^2$ [3]. Consequently, $\hat{A}[N]$ the estimate of A based on N observed data points, can be computed as a function of $\hat{A}[N-1]$ as

$$\hat{A}[N] = \hat{A}[N-1] + \frac{\frac{1}{\sigma_N^2}}{\sum_{n=0}^{N} \frac{1}{\sigma_n^2}} (x[N] - \hat{A}[N-1]). \tag{6}$$

If $x[n] = A + \overline{\epsilon}[n]$ where $\overline{\epsilon}[n]$ is a zero mean uncorrelated error, then the least square estimator is a best linear unbiased estimator (BLUE) [2]. Consequently

$$\operatorname{var}(\hat{A}[N-1]) = \frac{1}{\sum_{n=0}^{N-1} \frac{1}{\sigma_n^2}} \tag{7}$$

where $\operatorname{var}(\hat{A}[N-1])$ is the variance of $\hat{A}[N-1]$. Thus, in the generalized equation for calculating $\hat{A}[N]$ from $\hat{A}[N-1]$, the gain factor $k[N]$ in (7) is defined as

$$k[N] = \frac{\frac{1}{\sigma_N^2}}{\sum_{n=0}^{N} \frac{1}{\sigma_n^2}} = \frac{\operatorname{var}(\hat{A}[N-1])}{\operatorname{var}(\hat{A}[N-1]) + \sigma_N^2}. \tag{8}$$

Therefore, $\operatorname{var}(\hat{A}[N])$ can be represented as a function of $\operatorname{var}(\hat{A}[N-1])$ as given below [2]

$$\operatorname{var}(\hat{A}[N]) = (1 - k[N])\operatorname{var}(\hat{A}[N-1]). \tag{9}$$

Equation (6) can be rewritten as

$$\hat{A}[N] = \hat{A}[N-1] + k[N](x[N] - \hat{A}[N-1]) \tag{10}$$

where $\hat{A}[0] = 0$ and $\operatorname{var}(\hat{A}[0]) = \sigma_0^2$. Thus, $k[1]$ can be obtained from (8) and $\operatorname{var}(\hat{A}[N-1])$ from (9) followed by $\hat{A}[N]$ from (10). The simulation is stopped when the absolute difference of $\hat{A}[N]$ and $\hat{A}[N-1]$ is less than ϵ, which is a user defined limit for the error value.

The SLS method is a linear estimator. The computations are simple and are optimal if the input-power values are Gaussian in nature. In a linear estimation technique, the estimator $(\hat{\theta})$ is linear with respect to the observed data, $[x[N]]_{n=1}^{N}$ [2] or

$$\hat{A} = \sum_{n=0}^{N} a_n x[n] \tag{11}$$

where a_n are constants for $n = 0$ to $N-1$. The proof for convergence of the SLS algorithm can be found in [3].

B. Recursive Least Square Algorithm

In the SLS technique, the average power estimator is updated sequentially based on the previous value. In the RLS technique, the average-power estimator is updated based on a set of simulated values from a predefined number of previous iterations rather than a single value from the previous iteration as in the SLS. This helps in the faster convergence of the RLS algorithm. The algorithm is point estimate based, does not assume specific distribution characteristics of the input data, and the user can specify the accuracy level.

In the RLS estimation technique, there is a deviation in the given data $[x[k]]$, from the desired data $d[k]$ due to the characteristics of the input data where the desired data $d[k]$ closely matches the result. The power values (data) that are measured via experimentation (observed power values) have an inherent error when compared to the actual (desired) power values. The observed data is known as noisy data, a weighted combination of the observed data, is used to estimate the actual values. Let $[w_j(k)]_{j=0}^{N}$ denote the N weights to be computed and let $\overline{x}(k) = [x(k)x(k-1)\cdots x(k-N+1)]^T$ be a vector of N observed values. At the nth iteration, $\overline{x}(k)$ will be, $\overline{x}(k) = [x(n)x(n-1)\cdots x(n-N+1)]^T$. The coefficients $w_j(k)$, for $j = 0,\ 1,\ \ldots,\ N$, are adapted aiming at minimizing the objective function. The objective function for the adaptive RLS system is deterministically given as

$$\xi^d(k) = \sum_{i=0}^{k} \lambda^{k-1} e^2(i) \tag{12}$$

$$\xi^d(k) = \sum_{i=0}^{k} \lambda^{k-i} [d(i) - \overline{x}(i)\overline{w}(i)]^2 \tag{13}$$

where $e(i)$ is the output error at instant i and $\overline{w}(i) = [w_0(i)w_1(i)\cdots w_N(i)]$ the weight vector at instant i. The parameter λ is an exponential weighting function [2], which needs to be chosen in the range $0 \ll \lambda \leq 1$ (it is normally in the range 0.95–0.99). Differentiating (13) and solving for $w(k)$ we have

$$w(k) = R^{-1}(k)\, p_D(k) \tag{14}$$

where $R(k)$ and $p_D(k)$ are the correlation matrix of the observed data and the cross-correlation vector between the input data and the desired data, respectively. To compute the correlation matrix and its inverse, we apply the matrix inversion lemma, with $S(k) = R^{-1}(k)$, given as

$$S(k) = \frac{1}{\lambda}\left[S(k-1) - \frac{S(k-1)x(k)x^T(k)S(k-1)}{\lambda + x^T(k)S(k-1)x(k)}\right]. \tag{15}$$

Here, it is assumed that $R(k)$ is nonsingular. However, if $R(k)$ is singular, a generalized inverse should be used to obtain a solution for $\overline{w}(k)$. The cross-correlation matrix can now be calculated as

$$p_D(k) = \lambda p_D(k-1) + d(k)x(k). \tag{16}$$

The value for $d(k)$ is taken from the predicted value in the previous iteration, which is the average value till the kth iteration. The average power value is modeled as a (stationary) Gaussian random process and the time average is an unbiased (maximum likelihood) estimate for the

TABLE I
RESULTS FOR MCNC'91 CIRCUITS

Circuit	# of Iterations				Runtime (*ms*)		Power (*mw*)					% Error
Name	McPower [8]	DIPE [8]	SLS	RLS	SLS	RLS	SIM [8]	McPower	DIPE	SLS	RLS	$\frac{\lvert P_{RLS}-P_{SIM}\rvert}{P_{RLS}}$
c432	376	1386	221	187	44	58	1.646	1.570	1.650	1.637	1.658	0.72
c880	265	628	36	23	7	7	2.907	2.896	2.911	2.899	2.907	0.00
c1355	170	311	59	32	11	10	5.843	5.796	5.841	5.802	5.789	0.92
c1908	202	580	227	27	45	8	5.357	5.634	5.366	5.635	5.372	0.27
c2670	195	386	142	30	28	9	7.331	7.259	7.334	7.270	7.316	0.20
c3540	304	698	125	7	25	2	15.275	15.257	15.278	15.267	15.262	0.85
c5315	175	320	233	6	46	2	21.357	22.664	21.317	21.665	21.206	0.71
c6288	243	-	432	21	90	8	39.475	40.943	-	40.572	39.782	0.77
c7552	380	662	697	10	139	3	33.309	33.786	33.362	33.308	33.317	0.02
Avg Error												0.49

TABLE II
RESULTS FOR ISCAS'89 CIRCUITS

Circuit	# of Iterations			Runtime (*ms*)		Power (*mw*)				% Error
Name	McPower [4]	SLS	RLS	SLS	RLS	SIM	McPower	SLS	RLS	$\frac{\lvert P_{RLS}-P_{SIM}\rvert}{P_{RLS}}$
S27	289	57	24	10	8	125	124	125	124	0.80
S208	143	63	126	20	40	459	469	472	462	0.64
S298	97	108	82	30	26	819	826	826	829	1.20
S344	118	128	73	30	23	1024	1032	1023	1031	0.67
S349	111	173	54	40	17	1035	1040	1034	1041	0.57
S382	188	153	149	40	47	1132	1124	1127	1125	0.62
S386	168	55	35	10	11	1132	1142	1143	1145	1.13
S1494	274	97	24	30	8	3933	3954	4001	3973	1.00
S5378	366	259	112	70	36	12004	11901	11940	11894	0.92
S13207	1027	687	848	190	269	37748	37508	37508	37512	0.62
S15850	6103	4337	2076	1200	659	39985	40026	39992	40035	0.12
S35932	4649	4621	1157	1280	367	122048	123041	123049	123046	0.81
S38584	4905	5053	1998	1530	1400	112514	112605	112903	112623	0.09
Avg Error										0.7

mean of this random variable. So, we replace $d(k)$ by this time average. The estimator can then be calculated using the following

$$y(k) = \overline{w}^T(k)\overline{x}(k). \tag{17}$$

In a linear estimation technique, the estimator is the linear combination of the observed data

$$\xi^d = \sum_{i=0}^{k} \lambda w(k)x(k) \tag{18}$$

where λ is the forgetting factor, $w(k)$ is the coefficient/weight vector. The power values obtained through simulation, are the inputs for the RLS average power estimator. The estimator, the correlation matrix, and the cross-correlation vector values are calculated at each iteration using (17), (15), and (16), respectively, with the input data. The absolute difference between consecutive estimator values is computed after each iteration. If this value is less than the user defined value ϵ, (>0), the estimation is stopped. The resulting power value is computed as the average power dissipated in a circuit. The order of the algorithm is defined as the number of data points that are used in each iteration, toward achieving the objective. The forgetting factor λ, along with the order of the algorithm contribute to the speed and the accuracy of the algorithm. The minimum $\lambda_{\min}$ such that there is no loss of accuracy, is given in terms of the memory as

$$\lambda = 1 - \frac{1}{2N} \tag{19}$$

where N is the order of the algorithm. It should be noted that through experiments, we determine that the order of the algorithm to be 16 corresponding to $\lambda = 0.969$ from (19). The proof for convergence of the RLS algorithm can be found in [10].

III. Experimental Results

The two proposed methods were implemented using C on a Sun SPARC 4 workstation and tested for the MCNC'91 and the ISCAS'89 benchmark circuits. The input-power values were obtained from simulation using SIS, a system for sequential circuit synthesis [1] for random-input streams. The power dissipated corresponding to a set of input patterns was observed in SIS using the following setup: i) sampling, ii) zero delay, iii) uniform distribution for switching at the nodes, and iv) clock frequency of 20 MHz and power supply of 5 V. The ISCAS'89 circuits, were simulated with 32-bit input vectors in BDD mode.

The results of the proposed algorithms are compared with the Monte Carlo method (McPower) [4], as well as the distribution independent power estimator (DIPE) [8], a statistical method reported recently in the literature. Table I gives the number of iterations and the estimated average power for the MCNC'91 benchmark circuits using uncorrelated inputs as computed by simulation (denoted as SIM), McPower, DIPE, SLS, and RLS algorithms. It can be seen from the table, that the number of iterations for the RLS and the SLS algorithms are much lesser than that for the McPower and the DIPE. The RLS estimates and the DIPE estimates are within a range of 1%. The estimated power values using the RLS algorithm when compared with the simulated power values provided in [8], we see that the error is less than 1% of the simulated values. Similar results were seen for the ISCAS'89 circuits as shown in Table II. It should be noted that the sequential circuits were flattened and only the combinational part of the circuits were considered. The results corresponding to the accuracy level of 0.01 indicate the error values to be less than 1.2% with respect to simulation. We repeated the above experiments for higher accuracy levels of 0.001 and 0.0001 and the least square estimation algorithms still converge fast, while the Monte Carlo method does not converge at these levels.

IV. Conclusion

The SLS and RLS estimation methods proposed for average power estimation are linear, unbiased, and do not assume any distribution for the input sample data. The experimental results indicate that the least square estimation algorithms converge up to 24 times faster than the Monte Carlo and DIPE for random inputs.

Acknowledgment

The authors would like to thank the reviewers and the associate editor for their comments and suggestions.

References

[1] E. M. Sentovich *et al.*, "SIS: A system for sequential circuit synthesis," in *Electronics Research Laboratory*, Berkeley, CA, May 1992.

[2] S. M. Kay, *Fundamentals of Statistical Signal Processing—Estimation Theory*. Englewood Cliffs, NJ: Prentice Hall, 1993, ch. 7 and 8.

[3] P. S. R. Diniz, *Adaptive Filtering: Algorithms and Practical Implementation*. Norwell, MA: Kluwer, 1997.

[4] R. Burch, F. N. Najm, P. Yang, and T. Trick, "A Monte-Carlo approach for power estimation," *IEEE Trans. VLSI Syst.*, vol. 1, pp. 63–71, Mar. 1993.

[5] L. P. Yuan, C. C. Teng, and S. M. Kang, "Statistical estimation of average power dissipation in sequential circuits," in *Proc. Design Automation Conf.*, 1997, pp. 377–382.

[6] C. S. Ding, Q. Wu, C. T. Hsieh, and M. Pedram, "Statistical estimation of the cumulative distribution function for power dissipation in VLSI circuits," in *Proc. Design Automation Conf.*, 1997, pp. 371–376.

[7] M. G. Xakellis and F. N. Najm, "Statistical estimation of the switching activity in digital circuits," in *Proc. Design Automation Conf.*, 1994, pp. 728–733.

[8] L. P. Yuan, C. C. Teng, and S. M. Kang, "Statistical estimation of average power dissipation using nonparametric techniques," *IEEE Trans. VLSI Syst.*, vol. 6, pp. 65–73, Feb. 1998.

[9] C.-S. Ding, Q. Wu, C.-T. Hsieh, and M. Pedram, "Stratified random sampling for power estimation," *IEEE Trans. VLSI Syst.*, vol. 17, pp. 465–471, June 1998.

[10] A. K. Murugavel, N. Ranganathan, R. Chandramouli, and S. Chavali, "Average power in digital CMOS circuits using least square estimation," in *Proc. Int. Conf. VLSI Design*, 2001, pp. 215–220.

[11] R. Burch, F. N. Najm, P. Yang, and T. Trick, "McPOWER: A Monte Carlo approach to power estimation," in *Proc. Int. Conf. Computer-Aided Design*, 1992, pp. 90–97.

[12] T. L. Chou and K. Roy, "Accurate power estimation of CMOS sequential circuits," *IEEE Trans. VLSI Syst.*, vol. 4, pp. 369–380, Sept. 1996.

[13] C. Tsui, M. Pedram, and A. M. Despain, "Efficient estimation of dynamic power consumption under a real delay model," in *Proc. Int. Conf. Computer-Aided Design*, 1993, pp. 224–228.

[14] F. N. Najm, "Transition density: A new measure of activity in digital circuits," *IEEE Trans. Computer-Aided Design*, vol. 12, pp. 310–323, Feb. 1993.

[15] S. Bhanja and N. Ranganathan, "Dependency preserving probabilistic modeling of switching activity using Bayesian networks," in *Proc. Design Automation Conf.*, 2001, pp. 209–214.

[16] S. Gupta and F. N. Najm, "Power modeling for high-level estimation," *IEEE Trans. VLSI Syst.*, vol. 8, pp. 18–29, Feb. 2000.

[17] N. Shanbhag, "Lower bounds on power dissipation for DSP algorithms," in *Proc. Int. Symp. Low Power Design*, 1996, pp. 43–48.

[18] F. N. Najm and M. Y. Zhang, "Extreme delay sensitive and the worst case switching activity in VLSI circuits," in *Proc. Design Automation Conf.*, 1995, pp. 623–627.

[19] D. Marculescu, R. Marculescu, and M. Pedram, "Information theoretic measures for power analysis," *IEEE Trans. Computer-Aided Design*, vol. 16, pp. 599–610, June 1996.

[20] V. Krishna, R. Chandramouli, and N. Ranganathan, "Computation of lower and upper bounds for switching activity using decision theory," *IEEE Trans. VLSI Syst.*, vol. 7, pp. 125–129, Mar. 1999.

Locally Clocked Pipelines and Dynamic Logic

Gregg N. Hoyer, Gin Yee, and Carl Sechen

***Abstract*—Micropipelines and most of its variants use a delay-insensitive controller to moderate a pipeline. In search of improved performance, we depart from the delay-insensitive model in favor of a bounded-delay model for the controller. In particular, we demonstrate how a general delay-insensitive controller for level-sensitive pipelines can be improved by assuming a bounded-delay model and taking advantage of delay information to make the controller faster and more efficient. The new control scheme is referred to as locally clocked (LC) control. A highly pipelined logic technique called LC dynamic logic is presented that combines the bounded-delay controller with a latching dynamic logic gate design. Simulations comparing LC control with its delay-insensitive counterpart are presented. Also, an 8 × 8 bit multiplier with a maximum frequency of 715 MHz for a 1 μm CMOS process that uses LC dynamic logic is presented.**

***Index Terms*—Asynchronous pipelines, digital complementary metal–oxide–semiconductor (CMOS), dynamic-logic circuit, high performance, locally clocked (LC) control, multiplier design.**

I. Introduction

Asynchronous techniques avoid many of the difficulties associated with synchronous designs by using some type of internal mechanism to determine the system states [1]. One such asynchronous technique is micropipelines, which were introduced in Ivan Sutherland's Turing Award paper [2]. Micropipelines combine a delay-insensitive control structure and a bounded-delay pipelined datapath. In Section II, we provide an introduction to micropipelines and discuss a basic micropipeline scheme for controlling "traditional" linear pipelines that use level-sensitive storage elements. In Section III, the general micropipeline structure is modified, producing a new control scheme called locally clocked (LC) control. LC control abandons a strict delay-insensitive handshake protocol in favor of a bounded-delay model that enables the controller to achieve a higher level of performance and become more efficient. Section IV discusses a high-speed dynamic gate design that combines logic functionality and latching capabilities into one gate. The LC control scheme can be used in conjunction with these gates to provide a low-latency high-throughput circuit solution referred to as LC dynamic logic. Section V contains simulation results comparing the performance of the LC controller to its delay-insensitive counterpart. Finally, an 8 × 8 bit multiplier using LC dynamic logic that has a maximum frequency of 715 MHz for a 1-μm CMOS process is presented.

II. Micropipelines

Micropipelines are event-driven elastic pipeline structures that use a bundled data interface to communicate and transfer data between adjacent blocks. They are created by cascading stages that all adhere to the same handshake protocol. Sutherland's micropipeline [2] uses a delay-insensitive handshake protocol for the control circuitry that moderates the bounded-delay datapath. Since their introduction, many other micropipeline variants have been developed [3]–[5]. Fig. 1 shows

Manuscript received May 12, 1999; revised October 25, 1999. This work was supported by the Semiconductor Research Corporation, CDADIC, and Sun Microsystems.

The authors are with the Department of Electrical Engineering, University of Washington, Seattle, WA 98195 USA (e-mail: gnhoyer@ee.washington.edu; gsyee@ee.washington.edu; sechen@ee.washington.edu).

Publisher Item Identifier S 1063-8210(02)00793-X.