By case studies, we have shown that system-level specifications can be mapped systematically to an optimized block diagram (case study "saw-tooth generator"). Furthermore, we could systematically and in a quantitative way evaluate the effect of oversampling to a system (case study "converter"). In another case study, we could evaluate the influence of a different partitioning (analog/SC/digital) to the design. The proposed methods can be seen as a prototype of a first tool for "analog/digital codesign" because they permit the system-level optimization of mixed-signal applications *across the border* between analog and digital domains.

REFERENCES

[1] B. A. A. Antao, "Trends in CAD of analog IC's," *IEEE Circuits Devices Mag.*, pp. 31–41, Sept. 1996.

[2] C. Ouslis and A. S. Sedra, "Designing custom filters," *IEEE Circuits Devices Mag.*, vol. 11, no. 3, pp. 29–37, May 1995.

[3] B. A. A. Antao and A. J. Brodersen, "Techniques for synthesis of analog integrated circuits," *IEEE Design Test Comput.*, pp. 8–18, Mar. 1992.

[4] S. Donnay, K. Swings, G. Gielen, W. Sansen, W. Kruiskamp, and D. Leenaerts, "A methodology for analog design automation in mixed-signal ASIC's," in *Eur. Design Automation Conf. (EURO-DAC)*, Paris, France, Feb. 1994, pp. 530–534.

[5] A. Günter, *Wissensbasiertes Konfigurieren—Ergebnisse aus dem Projekt PROKON*, 1995.

[6] C. Grimm and K. Waldschmidt, "Repartitioning and technology-mapping of electronic hybrid systems," presented at the Design, Automation and Test in Europe '98 (DATE), Paris, France, Feb. 1998.

[7] C. Grimm, "Hybride Datenflussgraphen und ihre Anwendung beim Entwurf Analog/Digitaler Systeme," Ph.D. dissertation, J. W. Goethe-Universität Frankfurt am Main, VDI-Verlag, Germany, 2000.

[8] G. Stefanescu, "Reaction and control I. Mixing additive and multiplicative network algebras," *Logic J. IGPL*, vol. 6, no. 2, pp. 349–368, 1998.

[9] T. Stauner and C. Grimm, "Prototyping of hybrid systems—From hycharts to hybrid data-flow graphs," in *Proc. WDS'99 Electronic Notes in Theoretical Computer Science 28*, 1999.

[10] G. de Micheli, *Synthesis and Optimization of Digital Circuits*. New York: McGraw-Hill, 1994, p. B.6-44.

[11] A. V. Aho, R. Sethi, and J. D. Ullman, *Compilers—Principles, Techniques, Tools*. Reading, MA: Addison-Wesley, 1985.

[12] P. Oehler, "Wissensbasierte Konstruktion Gemischt Analog/Digitaler Systeme," Ph.D. dissertation, J. W. Goethe-Universität Frankfurt am Main, VDE-Verlag, Germany, 2000.

[13] R. J. van de Plassche, *Integrated Analog-to-Digital and Digital-to-Analog Converters*. Norwell, MA: Kluwer, 1994.

ENPCO: An Entropy-Based Partition-Codec Algorithm to Reduce Power for Bipartition-Codec Architecture in Pipelined Circuits

Shanq-Jang Ruan, Edwin Naroska, Yen-Jen Chang, Feipei Lai, and Uwe Schwiegelshohn

***Abstract*—This brief proposes a new algorithm to synthesize low power bipartition-codec architecture for pipelined circuits. The bipartition-codec architecture has been introduced as an effective power reduction technique for circuit design. The entropy-based partition-codec (ENPCO) algorithm extends this approach as it optimizes for both: power and area. It uses entropy as a criterion to balance between power and area. The ENPCO algorithm is composed of two phases: first, it clusters the output vectors with high occurrence into a group, moving all remaining output vectors into another group. The first group will be encoded in order to save power. Secondly, based on circuit entropy, output patterns are moved between both groups in order to balance power consumption and area overhead. A number of Microelectronic Center of North Carolina (MCNC) benchmarks were used to verify the effectiveness of our algorithm. Results demonstrate that ENPCO algorithm can achieve low power with less area overhead than the single-phase algorithm introduced in [1].**

***Index Terms*—Entropy, low power.**

I. INTRODUCTION

In modern VLSI chip designs, power dissipation has become one of the major concerns (besides area and speed) due to the widespread demand for high performance computing in portable systems. Because of its high throughput, pipelining is popularly used in such systems. Unfortunately, the registers that are inserted into the circuit to separate pipeline stages are a major source of power consumption [2]. Therefore, increasing system clock frequency without sacrificing low power properties requires advanced pipelined synthesis algorithms.

In today's VLSI circuits, the dominant fraction of average power dissipation is attributed to dynamic power dissipation caused by the switching activity of gate outputs [3]. As a result, many power optimization techniques that try to reduce switching activities to the minimum at various design levels have been proposed in recent years ([4], [5], [3] are good surveys to previous research efforts).

For logic level designs, two major low power techniques, *precomputation* and *gated-clock* are often applied. Alidina *et al.* first proposed a *precomputation-based* scheme, which selectively disables the inputs of a sequential logic circuit to achieve low power consumption [6]. Another precomputation scheme is circuit partitioning. The idea of partition for low power is that in behavioral descriptions of hardware, a small set of computation often accounts for most of the computational complexity as well as power dissipation. The authors of [7] extract CCC's (Common-Case Computation) during the design process and simplify the CCCs in a stand alone manner to achieve power saving.

Manuscript received December 26, 2001; revised April 19, 2002 and June 15, 2002.

S.-J. Ruan is with the Department of Electrical Engineering, National Taiwan University, Taipei 106, Taiwan, R.O.C. (email: stj@orchid.ee.ntu.edu.tw).

E. Naroska and U. Schwiegelshohn are with the Computer Institute Engineering, University of Dortmund, Dortmund 44221, Germany (email: edwin@ds.e-technik.uni-dortmund.de; uwe@ds.e-technik.uni-dortmund.de).

Y.-J. Chang is with the Department of Computer Science and Information Engineering, National Taiwan University, Taipei 106, Taiwan, R.O.C. (email: ychang@bulls.csie.ntu.edu.tw).

F. Lai is with the Department of Electrical Engineering and Department of Computer Science and Information Engineering, National Taiwan University, Taipei 106, Taiwan, R.O.C. (email: flai@cc.ee.ntu.edu.tw).

Digital Object Identifier 10.1109/TVLSI.2002.808422

In 1999, Ruan *et al.* proposed a bipartition-codec architecture which treats each output value of a combinational circuit as one state of a FSM, so that the most transitive states will be extracted to build a small subcircuit [1]. Additionally, Choi and Hwang partition a combinational circuit into multiple subcircuits through the recursive application of Shannon expansion with respect to the selected input variables [8]. While these approaches could successfully reduce power consumption they also incur some area overhead. Moreover the relationship between the number of partitions and power/area is still unclear. Ruan *et al.* showed that partitioning circuits with more than two-ways often does not save power due to the overhead of duplicated input latches and output multiplexors [9]. Benini *et al.* proposed a *gated-clock* approach to build low power FSMs [10]. The approach eliminates unnecessary glitches in the idle states of FSMs. In [11] and [12] a FSM is decomposed into a number of coupled submachines so that most of the time state transitions of high probability will be confined to smaller submachines.

Additionally, power estimation plays a significant role in low power design as it helps rating different design alternatives with respect to power consumption. Power estimation is especially important at higher level of abstraction in order to provide synthesis algorithms with information that can be used to optimize for low power. Various approaches for power estimation have been developed. Taking entropy-based estimation as example, it is an interesting solution to the problem of how to predict the area and power dissipated by a digital system for which an input/output description is available [13]–[16].

In this paper, we propose an algorithm called *entropy-based partition-codec algorithm* (ENPCO) which optimizes pipelined circuits for low power while incurring only small area overhead in most cases. Power saving is achieved by generating a bipartition-codec architecture from the original circuit. The ENPCO algorithm includes two phases: given a circuit described by PLA format, we first bipartition the PLA into two sub-PLAs. The first sub-PLA only includes patterns that frequently occur on the outputs (note that we assume that each *input* pattern have equal probability) while the other PLA is built from the remaining patterns. In the second phase, encoding is applied on the highly active sub-PLA and entropy is used to estimate the area of the entire architecture in order to fine tune both sub-PLAs (i.e., patterns are moved from one sub-PLA to the other in order to decrease area overhead). ENPCO algorithm decreases are overhead from 45.3% to 31.8% compared to the single-phase algorithm proposed in [17] while achieving almost the same power saving effects.

We illustrate our synthesis flow in Fig. 1. *Bipartition* is the first phase of ENPCO algorithm while *Fine Tune* and the remaining processes belong to second phase. In order to verify the results, our synthesis flow from the PLA specification to the transistor-level implementation and an accurate switch-level power estimation tool, EPIC powermill[1] , is used to estimate power consumption. The experimental results prove that ENPCO algorithm can save power consumption for most circuits of the MCNC benchmarks.

The major difference between bipartition-codec and precomputation architectures is that precomputation only disables some of the input pins to reduce switching activity of the combinational logic. However, the remaining input signals may also incur redundant switching activity in the entire combinational logic. Furthermore, precomputation does not account for the power dissipation of pipeline registers. Conversely, bipartition-codec architecture separates the combinational logic which ensures they will not influence each other. Moreover we take power dissipation of pipeline registers into account by applying the codec structure. This helps reducing power dissipation significantly.

The paper is organized as follows. Section II presents the basic models of power dissipation and area complexity of CMOS circuits. Section III describes the bipartition-codec architecture which is proposed in [1]. In Section IV, we formulate the problem and propose the ENPCO algorithm. Section V demonstrates the experimental results to verify and prove the feasibility of our algorithm. We conclude in Section VI.

[1]EPIC Powermill was developed by DPIC Design Technology, Inc.

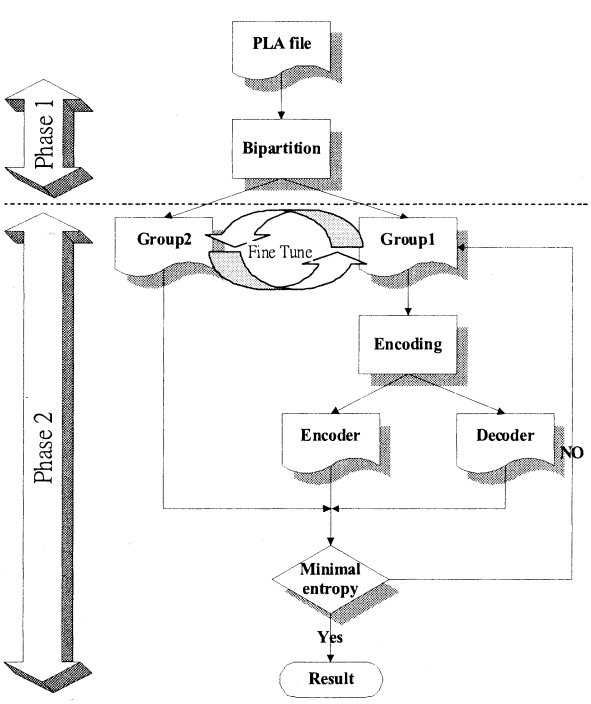

Fig. 1. Synthesis flow.

II. Preliminaries

In this section, we first describe the power dissipation model of CMOS circuits [4]. This leads to a relationship between power consumption and switching activity. Moreover, we describe an area estimation model based upon the input/output entropies [16].

A. Power Dissipation Model

In a digital CMOS circuit, the major source of power dissipation is switching activity so that the power consumption in digital CMOS circuits can be simply written as follows [4]:

$$P = 0.5 \times C_L \times V_{dd}^2 \times E(sw) \times f_{clk}, \tag{1}$$

where C_L is the loading capacitance, V_{dd} denotes the supply voltage, and f_{clk} is the clock frequency. These three parameters are primarily determined by the fabrication technology and circuit layout considerations such as transistor sizing. $E(sw)$ is the average transition number per clock cycle (referred to as the transition density), which can be determined by evaluating the logic function and the statistical properties of the input vectors. Obviously, (1) relates the dominant power dissipation to the switching activity of a CMOS circuit.

B. Area Estimation Model

For efficiency consideration, we adopt the area complexity model proposed by Cheng *et al.* [16]:

$$L(n, d, X) = (1 - d) \cdot k \cdot 2^n \cdot H(X) \tag{2}$$

where n is the number of input pins, d is the fraction of *don't care* terms in the truth table of the circuit, k is constant, and $H(X)$ is the entropy of the outputs. Let P be a set of probabilities $P = \{p(s_1), p(s_2), \ldots, p(s_{2^m})\}$ where $p(s_i)$ is a *probability*

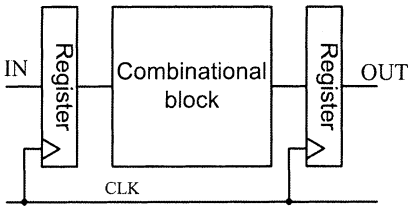

Fig. 2. Combinational pipelined circuit.

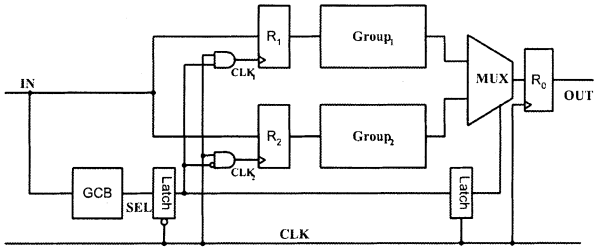

Fig. 3. Bipartition architecture.

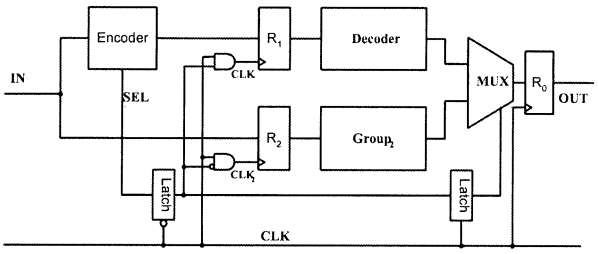

Fig. 4. Bipartition-codec architecture.

function $p : S \longrightarrow \{0..1\}$ which returns the frequency of an output pattern s_i. For example,

$$p(s_i) = \frac{x_i}{2^n}$$

where x_i is the number of appearances of pattern s_i at the outputs of the circuit if all possible 2^n input patterns are assigned to the circuit exactly once (note that some x_i may be 0). Then, the entropy $H(X)$ is defined as follows:

$$H(X) = \sum_{i=1}^{2^m} p_i \cdot \log_2 \frac{1}{p_i} \tag{3}$$

where m is the number of output pins of the circuit. Note that entropy is intuitively related to switching activity. For example, a n-input signal that rarely toggles suggests that the word-level values are relatively stagnant and many values will probably not appear. This skew occurrence probability gives a low entropy measure. Conversely, if signals are highly active, all word-line values are very likely to appear with the same probability. This maximizes entropy of the signals [18].

III. Bipartition-Codec Architecture

A pipeline stage can be represented by a combinational logic block separated by distinct registers as shown in Fig. 2.

Fig. 3 is a bipartition architecture without codec, where GCB selects a group to be enabled. Only one group can be active in a cycle. After bipartitioning a circuit, we apply the encoding technique introduced in [19] to the highly active subcircuit ($Group_1$). To reduce power consumption of the registers of the highly active subcircuit ($Group_1$), we replace it with a codec structure that consists of an encoder and a decoder (see Fig. 4). Note that the encoder not only encodes the outputs with minimal Hamming distance but also generates a *SEL* signal to choose the data path to be activated. In detail, the bipartition-codec architecture operates as follows. The input IN feeds into the Encoder and the registers of $Group_2$. Signal *SEL* and the encoding outputs become valid before the rising edge of the global clock *CLK*. If $SEL = 1$, the gated clock CLK_1 will be activated and CLK_2 be stopped when the next rising edge of *CLK* arrives. At this moment the encoding output passes through register R_1 and propagates into the decoder. Note that the input *IN* will not propagate through R_2. Hence the Decoder is selected to compute the outputs while $Group_2$ is idle.

Note that the presence of the low-enabled latch is needed for a correct behavior, because *SEL* may have glitches that must not propagate to the *AND* gate when *CLK* is high. The high-enabled latch memorizes the *SEL* function during a period of the global clock so that the multiplexer can select a correct output. The two latches work as a master-slave flip-flop. The interested reader may refer to [1] for further analysis of the architecture.

Note that the performance of the bipartition-codec architecture depends on the selection of output vectors that are assigned to Group_1 (encoder/decoder). Unfortunately, for n different output patterns there are $\binom{n}{1} + \binom{n}{2} + \cdots + \binom{n}{n-1} = 2^n - 2$ possible bipartitioning configurations. As a result, testing all configurations becomes computational expensive for large n.

IV. ENPCO Algorithm

A. Problem Formulation

The total area estimation model is

$$\begin{aligned}\text{Total\_Area} =& area_E + area_D + area_{G2} + overhead \\ =& L_E(n_E, d_E, X_E) + L_D(n_D, d_D, X_D) \\ &+ L_{G2}(n_{G2}, d_{G2}, X_{G2}) + overhead \\ =& L_E(n, d_E, X_E) + L_D(n_D, d_D, X_D) \\ &+ L_{G2}(n, d_{G2}, X_{G2}) + overhead\end{aligned} \tag{4}$$

where $area_E$, $area_D$ and $area_{G2}$ represent the area of Encoder, Decoder and $Group_2$ after being synthesized by SIS, respectively. The last term *overhead* consists of latches, logic gates, registers and multiplexers shown in Fig. 4. Note that we replaced n_E and n_{G2} with n because the numbers of input pins for *Encoder* and $Group_2$ are the same as the number of input pins of the original circuit.

For our optimization approach we use a slightly simplified version of (4) to estimate area consumption of the bipartition codec architecture by neglecting the *overhead* term, which is a constant throughout the optimization approach:

$$\begin{aligned}\text{Total\_Area}' =& L_E(n, d_E, X_E) + L_D(n_D, d_D, X_D) \\ &+ L_{G2}(n, d_{G2}, X_{G2}) \\ =& k \cdot 2^n((1 - d_E)H(X_E) + (1 - d_{G2})H(X_{G2})) \\ &+ k \cdot 2^{n_D}(1 - d_D)H(X_D).\end{aligned} \tag{5}$$

B. Two Phases ENPCO Algorithm

The input of the algorithm is the set of different output patterns $S = \{s_1, s_2, s_3, \ldots s_q\}$ of the circuit (note that here only pattern that actually appear at least once on the outputs are in S). Its output is a set of output pattern $Q \subset S$ that are assigned to the encoder/decoder part of the bipartition architecture.

PROCEDURE SelectCandidates(S, p)
  Input: (randomly ordered) set of output states S,
    probability function p
  Output: candidate set C
  Initialization: $C = \emptyset, prob = 0$;
BEGIN
  Sort $S = (s_1, s_2, \cdots s_q)$ by decreasing $p(s_i)$;
  WHILE $prob \leq 0.5$ **OR** p(first element in S) $> \frac{1}{q}$ **LOOP**
    $s = \{$first element in $S\}$;
    $C = C \cup \{s\}$;
    $prob = prob + p(s)$;
    $S = S \setminus \{s\}$;
  END LOOP;
  RETURN C;
END PROCEDURE;

Fig. 5. Algorithm to select candidates (phase 1).

PROCEDURE FindLowPowerSet(S, C)
  Input: randomly ordered output set S and candidate set C
  Output: low power set Q
  Initialization: $T = \emptyset, Q = \emptyset$
BEGIN
  WHILE $C \neq \emptyset$ **LOOP**
    select $s \in C, s \neq \emptyset$ so that $Total\_Area''(S, T \cup \{s\}) =$ min;
    $T = T \cup \{s\}$;
    $C = C \setminus \{s\}$;
    IF $Q = \emptyset$ **OR**
      $Total\_Area''(S, Q) > Total\_Area''(S, T)$ **THEN**
      $Q = T$;
    END IF;
  END LOOP;
  RETURN Q;
END PROCEDURE;

Fig. 6. Algorithm to select final set Q from C (phase 2).

The algorithm is divided into two phases. In the first phase we select a set of candidates $C \subset S$ which might be moved to the encoder/decoder part of the architecture. These candidates are selected by their frequency of appearance $p(s_i)$ on the outputs of the circuit. The motivation behind this approach is to assign a few highly active output pattern to the encoder/decoder section of the bipartition circuit in order to keep the small (with respect to area) part of the architecture active most of the time while the bigger part is mostly idle.

The second phase selects from the set of candidates C a subset of pattern $Q \subseteq C$ that are finally assigned to the encoder/decoder part.

Phase 1: Fig. 5 shows the algorithm used for Phase 1. In order to select a "promising" set of candidates for the second phase of our algorithm, C is determined by two main rules. First, all patterns that have a probability above the average value $1/q$ are added to candidate set C. If the sum of probabilities of all patterns (*prob*) in C is less than 1/2 then further pattern are added (in decreasing probability order) until 1/2 is reached.

Phase 2: After determining the candidate set C, the next step is to select a subset of elements from C that will be assigned to the encoder/decoder section of our architecture. However, our selection does not only focus on reducing power but also on reducing the area required to implement the entire architecture. Hence, our goal is to find a nonempty set $Q \subseteq C$ that if assigned to the encoder/decoder part of our architecture results in a minimal total area complexity. Note that an exhaustive search is impractical as $O(2^{|Q|})$ permutations must be analyzed. Hence, we use a greedy approach to find an acceptable solution in $O(|Q|^2)$.

During runtime of the algorithm area consumption of various configurations are tested and compared as follows: based on a chosen pattern set $V \subseteq C$, the algorithm introduced in [19] is executed to find a good binary representation of V. The encoding results are then used to estimate area of the total circuit using the following Equation:

$$\begin{aligned}\text{Total\_Area}''(S, V) =& 2^n((1 - d_{E(V)})H(X_{E(V)}) \\ &+(1 - d_{G2(S\setminus V)}H(X_{G2(S\setminus V)}))) \\ &+2^{n\,D(V)}(1 - d_{D(V)})H(X_{D(V)}). \qquad (6)\end{aligned}$$

It is derived from (5) by removing constant k. V is the set of output states that shall be associated with the encoder/decoder part while S is the entire output set. Note that values $d_{E(V)}$, $X_{E(V)}$, $n_{D(V)}$, $d_{D(V)}$ and $X_{D(V)}$ depend on the results obtained from encoding.

The actual algorithm is shown in Fig. 6. It takes the candidate set C as a parameter and returns the final set Q to be assigned to the encoder/decoder part. The algorithm uses two internal sets: T is a temporary set of pattern while Q holds the best pattern set found so far.

In each iteration of the while loop a pattern $s \in C$ is selected so that the total area of the circuit is minimal. This is done by applying the encoding algorithm introduced in [19] on each combination $T \cup s$ and using (6) to *estimate* the area of the total circuit (i.e., area of $Encoder + Decoder + Group_2$). From these estimation results the best (smallest total area) is selected and the corresponding s is added to T and removed from C. Hence, in each iteration of the while loop a single pattern is removed from C and added to T until C is empty. Finally, in each iteration the area consumption of T is compared to the area value of Q to determine the best pattern set among all iterations executed so far.

In order to explain the algorithm in more detail a short example is given: suppose we have a candidate set $C = \{s_1, s_2, s_3, s_4\}$ that has been extracted from the output pattern set S (I.e., $C \subset S$). During the first iteration a pattern s is determined that gives us a minimum area value. For example, we determine Total_Area$''(S, s_1)$, Total_Area$''(S, s_2)$, Total_Area$''(S, s_3)$ and Total_Area$''(S, s_4)$ in order to find the best pattern. Let us assume that Total_Area$''(S, s_3)$ is the minimum among all these values. Hence, T^1 is set to $\{s_3\}$ and s_3 is removed from C (i.e., $C^1 = \{s_1, s_2, s_4\}$; the exponent denotes the iteration number). Next we compare Total_Area$''(S, \{s_3, s_1\})$, Total_Area$''(S, \{s_3, s_2\})$ and Total_Area$''(S, \{s_3, s_4\})$. Suppose that Total_Area$''(S, \{s_3, s_1\})$ is the minimal area among these three values. As a result, we get $T^2 = \{s_3, s_1\}$ and $C^2 = \{s_2, s_4\}$. Let us assume that in the next iteration pattern s_4 is selected giving $T^3 = \{s_3, s_1, s_4\}$ and $C^3 = \{s_2\}$. Finally, in the next iteration the last remaining pattern in C is added giving $T^4 = \{s_3, s_1, s_4, s_2\}$ and $C^4 = \emptyset$. Hence, we get four different configurations $T^1 = \{s_3\}$, $T^2 = \{s_3, s_1\}$, $T^3 = \{s_3, s_1, s_4\}$ and $T^4 = \{s_3, s_1, s_4, s_2\}$. Then, the set with a minimum area is returned as the final result. Note that the final result is not necessarily T^1. For example, adding a pattern to T^1 may *decrease* the complexity of the encoder/decoder part due to the fact that among the decoded/encoded value the selection signal SEL must be also generated by the encoder.

C. Complexity Analysis

The first phase of the ENPCO algorithm starts with a sort operation, which requires $O(q \log q)$ if an efficient sorting algorithm is applied (see Fig. 5). Then, the following while loop is executed q times. This gives an overall complexity of $O(q \log q)$ for the first phase.

Applying the encoding algorithm from [19] during phase 2 requires $O(|C|^3)$ steps, where $|C|$ is the size of the candidate set. Further, the encoding algorithm is executed $O(|C|^2)$ times as the while loop from procedure FindLowPowerSet (see Fig. 6) iterates $|C|$ times and $O(|C|)$

configurations are tested in each iteration. Hence, the overall complexity of phase 2 is $O(|C|^5)$.

V. Simulation Environment and Results

A. Simulation Environment

The ENPCO algorithm has been implemented in C++ on a SUN Sparc station and several MCNC benchmark circuits were used to test it. The rugged script of SIS [20] was used to synthesize *Encoder*, *Decoder*, and $Group_2$. Power dissipation estimation was done by EPIC PowerMill. We used 5 v supply voltage, and a clock frequency of 20 MHz.

In our experiments, we selected the primitive standard cells which were provided by CCL[2] in 0.8 μm technology. From the data book of CCL 0.8 μm standard cell library, we extracted the propagation delays and driving capabilities of every standard cell. These numbers were used to construct a technology file in genlib format to be used by SIS. Further, the standard cells in SPICE subcircuit format were converted into the EPIC format using EPIC utility *gentech*. So, as the final synthesized results are in EPIC format, EPIC PowerMill could be used to estimate power consumption of the pipelined circuits. The overall flow is shown in Fig. 7.

The MCNC benchmarks that are given in two-level PLAs are bipartitioned and encoded into three PLAs: *Encoder*, *Decoder*, and $Group_2$. A synthesis script, *script.rugged* of SIS was used to optimize each PLA. Then the library binding program *map* of SIS was used to generate a corresponding gate-level file in BLIF format (Berkeley Logic Interchange Format). A BLIF to SPICE format converter was used to convert the BLIF gate-level description to SPICE transistor-level according to a layout-extracted netlist file. Finally, the utility *spice2e* of EPIC converted the pipeline circuits from SPICE format to EPIC format.

B. Experimental Results

We assumed uniform probability distribution for the primary inputs of the pipelined circuit, but this assumption is not restrictive. For example, in a pipelined circuit the input probability distribution can be computed from the output probability of the previous stage. The registers of the output part are unchanged in our architectures. Hence, we do not consider the effect of these registers on area and power dissipation. The area unit and power unit are 128 μm^2 and μW in our experiments.

The area and power dissipation of the original, single-phase and ENPCO algorithms are tabulated in Table I. The "Original" column shows the number of inputs (*#I*), outputs (*#O*), area *Area* and power dissipation *Power* of each circuit. The "single-phase algorithm" and "ENPCO algorithm" columns show *Area*, *Power*, percentage of area increase *AI*% computed as $100(Area_{\text{bipartition}} - Area_{\text{original}})/Area_{\text{original}}$, and power reduction *PR*% computed as

$$\frac{100(Power_{\text{original}} - Power_{[\text{single-phase}|ENPCO]})}{Power_{\text{original}}}.$$

The column *DN* shows the number of output vectors that were assigned to the codec structure.

The ENPCO algorithm is capable of reducing the area of combinational parts (i.e., *Encoder*, *Decoder*, and $Group_2$). However, area overhead (i.e., two latches, two AND gates and multiplexers as well as the additional registers) may become significant if the original circuit is *small*. Further, these additional components also consume power. As a result, for small circuits, the area overhead introduced by our architecture may become significant while saving only little or no power.

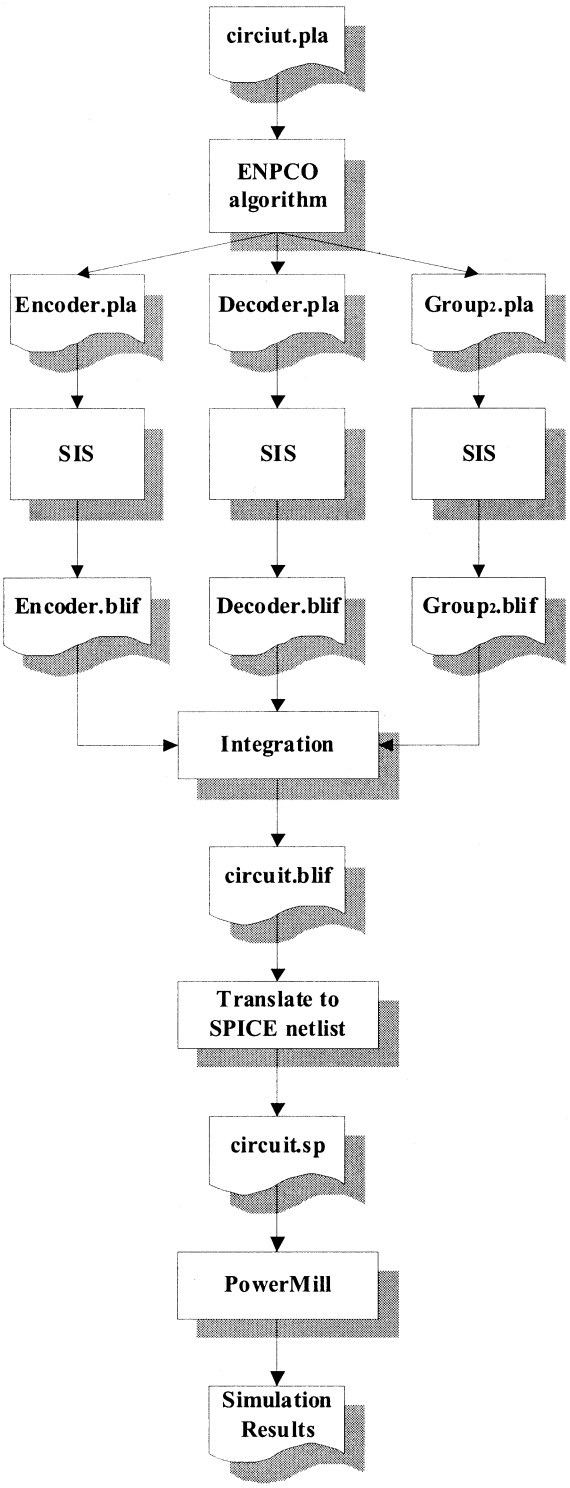

Fig. 7. Simulation flow.

In Figs. 8 and 9, we show the comparison of power reduction and area increase. In Fig. 8, the curve labeled *Power increase* is obtained by subtracting ENPCO algorithm's PR% from single-phase algorithm's PR%. Similar, *Area reduction* of the Fig. 9 is obtained by subtracting ENPCO algorithm's AR% from single-phase algorithm's AR%. As

[2]CCL stands for Computer and Communication Research Labs and is one of the members of Industrial Technology Research Institute in Taiwan.

TABLE I
SIMULATION RESULTS OF ORIGINAL CIRCUIT AND BIPARTITION ARCHITECTURES

Circuits	Original				single-phase Algorithm [1]				ENPCO Algorithm				
	#I	#O	Area	Power	Area	AI%	Power	PR%	DN	Area	AI%	Power	PR%
sao2	10	4	571	3361	725	27.0	1255	62.7	2	675	18.2	1392	58.6
con1	7	2	209	1506	384	83.7	1072	28.8	2	300	43.5	1176	21.9
misex1	8	7	340	2238	441	29.7	1555	30.5	2	399	17.4	1474	34.1
rd84	8	4	531	3176	774	45.8	2649	16.6	2	714	34.5	2920	8.1
rd73	7	3	370	2362	740	100.0	2230	5.6	3	605	63.5	2316	1.9
table5	17	15	3203	7946	4312	34.6	5748	27.7	2	4145	29.4	6831	14.0
cm85a	11	3	383	2535	312	-18.5	1757	30.7	3	312	-18.5	1757	30.7
cm163a	16	5	475	3573	716	50.7	2688	24.8	1	716	50.7	2688	24.8
t	5	2	132	1007	244	84.8	477	52.6	2	193	46.2	501	50.3
cu	14	11	485	3144	637	31.3	1470	53.2	3	593	22.3	1538	51.1
C17	5	2	132	1005	223	68.9	519	48.3	1	193	46.2	500	50.2
x2	10	7	371	2442	533	43.7	1646	32.6	3	482	29.9	1919	21.4
rd53	5	3	216	1499	395	82.9	1067	28.8	1	356	64.8	1112	25.9
cm162a	14	5	444	3109	654	47.3	2369	23.8	3	654	47.3	2369	23.8
cmb	16	4	440	3265	435	-1.1	811	75.2	1	435	-1.1	811	75.2

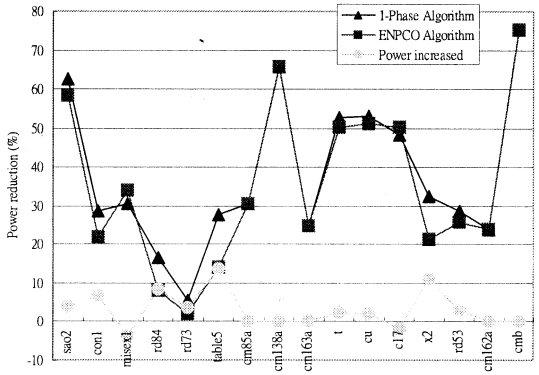

Fig. 8. Power reduction rate (%) for single-phase and ENPCO algorithms applied to MCNC benchmarks.

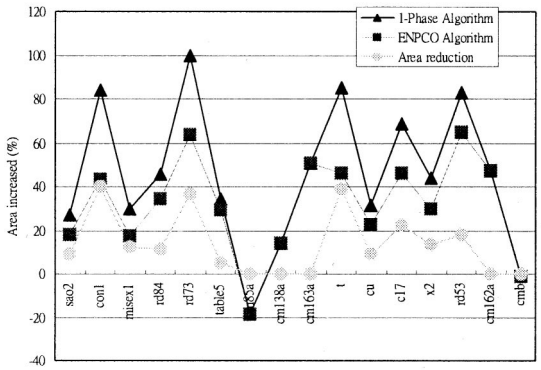

Fig. 9. Area increased rate (%) for single-phase and ENPCO algorithms applied to MCNC benchmarks.

shown in Figs. 8 and 9, ENPCO algorithm has almost the same power saving results compared to the single-phase algorithm but requires less area overhead. Fig. 10 shows the AI% and PR% values obtained by ENPCO algorithm for comparison.

Table II compares the average area and power reduction of previous schemes with our proposed algorithm. The *precomputation* column represents the circuits implemented by precomputation-based method [6]. The *Choi's algorithm* and *Choi's area constraint* columns represent the circuits implemented by Choi's algorithm and Choi's algorithm with area constraint, respectively [8]. The data shown in these three columns are cited from [17]. The *Single-phase* column represents the circuits implemented by a single-phase algorithm introduced in [1]. Our proposed algorithm is shown in column *ENPCO*. Obviously, ENPCO algorithm obtained more power reduction and less area overhead than Choi's algorithm with area constraint. Moreover, ENPCO algorithm achieves almost the same power saving as the single-phase algorithm, however, it adds significant less area overhead. Although precomputation and Choi's area constraint algorithms can achieve power reduction and reduce the area marginally, their power reductions are limited. In summary, our proposed architecture obtained significant power saving with less area overhead compared to previously published techniques.

Our approach reduces power consumption from the circuits with few high-probability output vectors. We prove the bipartition-codec circuit consumes less power than the original one. The power dissipation of

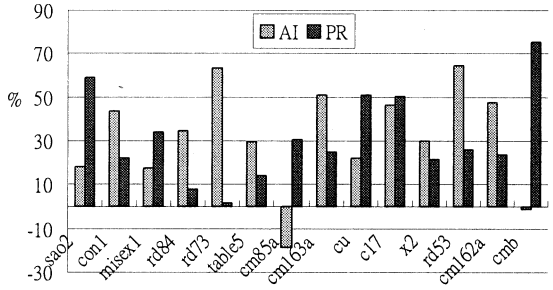

Fig. 10. Area increase (AI%) and power decrease (PR%) comparison for ENPCO algorithms applied to MCNC benchmarks.

the input registers is also reduced for the sake of less switching of input variables. Further, the codec architecture not only reduces the switching activity of the input registers but also the internal switching activity of *Decoder*.

However, there are circuits that are not suitable for bipartition-codec methodology. Consider the benchmark circuits in Table I. We observed that for some circuits such as *sao2 misex1* and *cmb*, the ENPCO algorithm reduces power up to 75.2%. However, for other circuits such as *rd73*, it provides only limited power reduction. Moreover, the area overhead introduced by the architecture is significant. It seems that these circuits are not suitable for this approach.

TABLE II
AVERAGE AREA AND POWER COMPARISON AMONG DIFFERENT METHODOLOGIES

	Precomputation	Choi's algorithm	Choi's area constraint	Single-phase [1]	ENPCO
Area (%)	-0.5	99	-0.3	45.3	31.8
Power (%)	-9.6	-32	-11.6	-38.0	-34.9

TABLE III
OUTPUT PATTERNS LIST AND ITS CORRESPONDING OCCURRENCES ON SOME BENCHMARK CIRCUITS

sao2(58.6%)		cmb(34.1%)		rd73(1.9%)	
Output patterns	Occurrences	Output patterns	Occurrences	Output patterns	Occurrences
0000	513	0110	65489	110	35
0010	257	0011	16	001	35
0011	219	0100	15	100	21
0100	8	1110	15	011	21
1000	7	1100	1	010	7
1100	6			101	7
0001	5			000	1
0101	4			111	1
1001	3				
1101	2				

Hence, if we find a method to detect these kind of circuits in advance we could avoid applying our technique on these cases saving synthesis runtime. Table III displays the output patterns for a subset of the benchmark circuits. While for models *sao2* and *cmb* few output patterns are activated by many different input patterns, model *rd73* shows a more balanced output pattern probability. Hence, for models *sao2* and *cmb* good power reduction results could be achieved while our approach fails for *rd73*. For example, our approach is promising for models where most input patterns are mapped to a small set of output patterns. For other circuits the power reduction that can be achieved is often small while the area overhead is significant. Hence, for these kind of circuits it is better to avoid applying our algorithm.

In order to detect these circuits, we introduce an extra check in the proposed algorithm. Consider a set of q output pattern $W = \{s_1, s_2, \ldots, s_q\}$ where each of these pattern appear at least once on the output. Then, $Y = \{y_1, y_2, \ldots, y_q\}$ is the frequency vector of W, where y_i is the number of appearences of pattern s_i at the output. Y meets the following two conditions (n is the number of input pins):

$$y_i \in \{1, 2, \ldots, (2^n - q + 1)\} \tag{7}$$

and

$$\sum y_i = 2^n. \tag{8}$$

Hence, the mean value of Y is $\overline{Y} = 2^n/q$. In order to rate a circuit we calculated the normalized deviation ($C.V.$)

$$C.V. = \frac{\sqrt{\frac{1}{q}\sum_{i=1}^{q}(y_i - \frac{2^n}{q})^2}}{S_{\max}} \tag{9}$$

where the numerator is the standard deviation of Y and $S_{\max}$ is the maximum standard deviation value among all possible sets Y that meet (7) and (8). Hence, $C.V.$ holds $0 \leq C.V. \leq 1$. The deviation of Y is maximal if all except one y_i are set to 1. E.g., y_1 may be set to $y_1 = 2^n - q + 1$ and all remaining y_i are set to $y_i = 1$. As a result, $S_{\max}$ is

$$S_{\max} = \sqrt{\frac{q-1}{q}\left(1 - \frac{2^n}{q}\right)^2 + \frac{1}{q}\left(2^n - q + 1 - \frac{2^n}{q}\right)^2}. \tag{10}$$

In the following, we prove that (10) is actually the maximum value.

Proof: For the proof it is sufficient to show that $S'(Y') = \sum_{i=1}^{q}(y_i - 2^n/q)^2$ is maximum for $Y' = (\hat{y}, 1, 1, \ldots, 1)$, where $\hat{y} = 2^n - q + 1$. Subtracting a value δ with $0 < \delta \leq \hat{y} - 1$ from y_1 and adding it to y_2 creates a new vector Y'':

$$Y'' = (\hat{y} - \delta, 1 + \delta, 1, \ldots, 1).$$

Note that δ must not be greater than $\hat{y} - 1$ to ensure that all elements of Y'' are greater or equal to 1. Calculating the difference $S'(Y') - S'(Y'')$ gives

$$S'(Y') - S'(Y'') = 2\delta(\hat{y} - 1 - \delta) \geq 0. \tag{11}$$

Hence, $S'(Y')$ is greater or equal to $S'(Y'')$. For example, decreasing y_1 and increasing y_2 by the same value does not increase S'. Based on the previous analysis we also conclude that decreasing y_2 and increasing y_3 of Y'' by δ' ($0 < \delta' \leq \delta$) also does *not* increase S'. Consequently, $S'(Y''') \leq S'(Y')$ is valid for each vector

$$Y''' = (\hat{y} - \delta_1, 1 + \delta_1 - \delta_2, 1 + \delta_2 - \delta_3, \ldots, 1 + \delta_{q-1}),$$

with $\hat{y} - 1 \geq \delta_1 \geq \delta_2 \geq \cdots \geq \delta_{q-1} \geq 0$. As for each valid Y an appropriate δ_i-set can be determined such that $Y = Y'''$, S' as well as the standard deviation are maximal for $Y = Y'$.

In order to characterize a circuit we evaluate (9). If $C.V.$ is large (suppose > 0.8), it implies most of the inputs are mapped to some few outputs. Hence the bipartition-codec methodology is capable of reducing power consumption significantly. If $C.V.$ is small (say ≤ 0.8), the occurrence of each output pattern is uniformly distributed. This implies that our bipartition-codec methodology will probably achieve

only little power saving but will introduce a significant area overhead. The threshold value (here: 0.8) can be changed to match different needs and applications. For example, if low power is the dominant consideration disregarding area overhead, we can use a smaller threshold value. If area and power reduction are both our optimization targets, we can use a larger threshold value to filter out the unsuitable circuits.

Scaling Problem: In [2], the authors indicate that according to the simulation results, the total power consumption of a design is similar among different technologies. From a theoretical viewpoint, a scaling factor SC defines the technology change in a certain physical parameter (e.g., gate oxide thickness) from one technology generation to another [21]. For instance, changing the channel length from 0.35 to 0.25 μm gives SC the value 0.71.

The ideal scaling for deep-submicrometer of power dissipation and area is SC^2. For the computation of AI% and PR% in our experiments, they are the same in different technologies. Therefore, the results we obtained and the conclusion we draw in this paper apply to other technology levels as well.

VI. Conclusion and Discussion

In this paper, we have proposed an effective ENPCO algorithm to minimize power dissipation and area overhead for pipelined circuits. We bipartition the circuits into two groups: one group contains patterns that occur often at the outputs which implies higher activity, the other group contains the remaining output patterns. We apply a codec structure to the highly active group for power saving. Then we estimate the area of combinational blocks (Encoder, Decoder and $Group_2$) by using an entropy based approach. Finally we choose the configuration with minimal estimated area overhead as our final synthesized result. We compared the single-phase and ENPCO algorithms by power and area. The experiments show that the ENPCO algorithm achieves a good trade-off between area overhead and power dissipation. The power dissipation benefit of the bipartition-codec architecture synthesized by our ENPCO algorithm comes from the following three reasons:

2) the lengths of the registers, which are used to store the output of each stage, are reduced after encoding;
3) the Hamming distance of the register values is smaller than before;
4) the circuit switching activity of the combinational block is reduced.

The circuit will benefit from our architecture if a small number of output vectors dominate most of the circuit behavior (e.g., see models *sao2* and *cmb* in Table III). Nevertheless, circuits that are characterized by a uniform output vector probability distribution are not suited to our architecture. For example, the output pattern distribution of *rd73* is more uniform than those of the others models (see Table III). As a result, power reduction is less significant than that of *sao2* and *cmb*.

Compared to precomputation architecture, precomputation only disables the partial input pins for reducing the switching activity of combinational logic. Hence, the remainder input signals may also incur redundant switching activity in the entire combinational logic. Furthermore, precomputation does not account for the power dissipation of pipeline registers. Conversely, bipartition-codec architecture not only separates the combinational logic to ensure that they will not influence each other but also reduces power dissipation of the pipeline registers by applying a codec structure.

References

[1] S.-J. Ruan, R.-J. Shang, F. Lai, S.-J. Chen, and X.-J. Huang, "A bipartition-codec architecture to reduce power in pipelined circuits," *Proc. IEEE/ACM Int. Conf. Computer Aided Design*, pp. 84–89, Nov. 1999.

[2] W. Ye and M. J. Irwin, "Power analysis of gated pipeline registers," in *Proc. Twelfth Annual IEEE International ASIC/SOC Conference*, 1999, pp. 281–285.

[3] L. Benini and G. D. Micheli, *Dynamic Power Management: Design Techniques and CAD Tools*. Norwell, MA: Kluwer Academic, 1998.

[4] M. Pedram, "Power minimization in IC design: Principles and applications," *ACM Trans. Design Automat. Electron. Syst.*, vol. 1, pp. 3–56, Jan. 1996.

[5] K. Roy and S. C. Prasad, *Low-Power CMOS VLSI Circuit Design*. New York: Wiley, 2000.

[6] M. Alidina, J. Monterio, S. Devadas, S. Devadas, A. Ghosh, and M. Papaefthymiou, "Precomputation-based sequential logic optimization for low power," *IEEE Trans. VLSI Syst.*, vol. 2, pp. 426–436, Dec. 1994.

[7] G. Lakshminarayana, A. Raghunathan, K. Khouri, N. Jha, and S. Dey, "Common-case computation: A high-level technique for power and performance optimization," in *Proc. Design Automat. Conf.*, 1999, pp. 56–61.

[8] I.-S. Choi and S.-Y. Hwang, "Circuit partition algorithm for low-power design under area constraint using simulated annealing," *Proc. Inst. Elect. Eng., Circuit Devices Syst.*, vol. 146, pp. 8–15, Feb. 1999.

[9] S.-J. Ruan, J.-C. Lin, P.-H. Chen, F. Lai, K.-L. Tsai, and C.-W. Yu, "An effective output-oriented algorithm for low power multipartition architecture," in *Proc. IEEE Int. Conf. on Electronics, Circuits and Systems*, 2000, pp. 609–612.

[10] L. Benini and G. D. Micheli, "Automatic synthesis of low-power gated clock finite-state machine," *IEEE Trans. Computer-Aided Design*, vol. 15, pp. 630–643, Sept. 1996.

[11] J. C. Monterio and A. L. Oliveira, "Finite state machine decompostion for low power," in *Proc. Design Automat. Conf.*, 1998, pp. 758–763.

[12] S.-H. Chow, Y.-C. Ho, T. Hwang, and C. L. Liu, "Low power realization of finite state machines— A decompostition approach," *ACM Trans. Design Automat. Electron. Syst.*, vol. 1, pp. 315–340, July 1996.

[13] A. Lioy, E. Macii, M. Poncino, and M. Rossello, "Accurate entropy calculation for large logic circouts based on output clustering," in *Proc. Seventh Great Lakes Symposium on VLSI*, 1997, pp. 70–75.

[14] M. Nemani and F. N. Najm, "High-level area and power estimation for VLSI circiuts," *IEEE Trans. Computer-Aided Design*, vol. 18, pp. 697–713, June 1999.

[15] E. Macii, M. Pedram, and F. Somenzi, "High-level power modeling, estimation, and optimization," *IEEE Trans. Computer-Aided Design*, vol. 17, pp. 1061–1079, Nov. 1998.

[16] K.-T. Cheng and V. D. Agrawal, "An entropy measure for the complexity of multi-output boolean functions," *Proc. 27th ACM/IEEE Design Automation Conference*, pp. 302–305, 1990.

[17] S.-J. Ruan, R.-J. Shang, F. Lai, and K.-L. Tsai, "A bipartition-codec architecture to reduce power in pipelined circuits," *IEEE Trans. Computer-Aided Design*, pp. 343–348, Feb. 2001.

[18] G. Yeap, *Practical Low Power Digital VLSI Design*. New York: Kluwer Academic, 1998.

[19] L. Benini and G. D. Micheli, "State assignment for low power dissipation," *IEEE J. Solid-State Circuits*, vol. 30, pp. 258–268, Mar. 1995.

[20] "SIS: A System for Sequential Circuit Synthesis," Univ. California, Dept. EE and CS, Electron. Res. Lab., Berkeley, CA, 1992.

[21] D. Sylvester and C. M. Hu, "Analytical Modeling and Characterization of Deep-Submicrometer Interconnect," *Proc. IEEE*, vol. 89, 2001.