A Variation-Aware TED-Based Approach for Nano-CMOS RTL Leakage Optimization

Shibaji Banerjee[1], Jimson Mathew[1], Saraju P. Mohanty[2*],

Dhiraj K. Pradhan[1], and Maciej J. Ciesielski[3]

[1]Dept. of Computer Science, University of Bristol, Bristol, BS8 1UB, UK, {shibaji, jimson, pradhan}@compsci.bristol.ac.uk

[2]Dept. of Computer Science and Engineering, University of North Texas, Denton, TX, 76202, USA, saraju.mohanty@unt.edu

[3] Dept. of Electrical & Computer Engineering, University of Massachusetts, Amherst, MA 01003, USA, ciesiel@ecs.umass.edu

* corresponding author:

Address:

University of North Texas

Department of Computer Science and Engineering

3940 N. Elm St., Room F201

Denton, TX 76207-7102, USA

Office : +1 (940) 565-3276

Fax : +1 (940) 565-2799

Email : saraju.mohanty@unt.edu

Date of Receiving: **to be completed by the Editor**

Date of Acceptance: **to be completed by the Editor**

A Variation-Aware TED-Based Approach for Nano-CMOS RTL Leakage Optimization

Shibaji Banerjee, Jimson Mathew, Saraju P. Mohanty,

Dhiraj K. Pradhan, and Maciej J. Ciesielski

Abstract — As the CMOS technology scales down to nanometer regime the process variations have profound effect on circuit attributes. Meeting timing and power constraints under such process variations in nano−CMOS circuit design is becoming increasingly difficult. A shifting from worst-case based analysis and optimization to statistical or probability based analysis and optimization at every level of circuit abstraction has happened. This paper presents a Taylor Expansion Diagram (TED) based approach for statistical optimization during high-level synthesis (HLS). A variation-aware simultaneous scheduling and resource binding algorithm is proposed which maximizes the power yield under timing yield and performance constraint. For this purpose, a multiple-oxide thickness (multi−T_{ox}) library at 45nm CMOS is characterized under process variation. The delay and power distribution of different functional units are accurately analyzed. The proposed variation-aware algorithm uses those components for generating low-power register-transfer level (RTL) descriptions under a given timing yield and performance constraint. The experimental results show significant improvement as high as 95% on leakage power yield under given constraints.

Keywords — Process Variation, Leakage Power, RTL optimization, Low-power synthesis

1 INTRODUCTION

As CMOS technology continues to scale down to achieve higher performance and higher level of integration, the process variation and power dissipation pose new and difficult challenges for integrated circuit designers. When the transistors are quite small, e.g., for a 45nm CMOS technology, the gate-oxide layer of 1.2nm is only a few SiO_2 atomic layers, the transistor parameters vary from die to die or even in the same die. In other words, each transistor in a die or wafer is different. In the context of nano−CMOS technology, the challenges for design engineers have significantly increased due to introduction of variability [1], [2], [3]. The scaling down of technology has resulted in significant deviations from the nominal values of transistor parameters, such as channel length, threshold voltage, and gate-oxide thickness. For example, variation in gate length increases from 35% in a 130nm technology to almost 60% in a 65nm technology which results in the large variation in leakage and performance of the designed circuit [1].

Design decisions are often based on the nominal values of power, and performance and on the assumption that all the transistors are alike across dies and wafers [4]. Thus, the design decisions based on the nominal models may not be accurate because the models are either overestimations or underestimations of actual values; hence, the resultant circuits may not be optimal. The transistor parameter variations may be due to several factors, including changes in oxide thickness, substrate, polysilicon, and implant impurity levels; surface charge; and lithographic process. To avoid process variation, worst-case based approaches are used for estimation of delay and power consumption. However, these lead to include too much pessimism to the design [5]. To tackle this problem different Statistical Static Timing Analysis (SSTA) techniques have been developed at the gate level for delay and power calculation. However, only a few SSTA techniques are available at the higher level of circuit abstraction (e.g. register-transfer level or RTL). Tackling process variation and power dissipation issues at a higher level of the design hierarchy will lead to high yield circuits at minimal design cost. Thus, analysis and optimization at the higher level of the design abstraction is required to tackle process variability and power dissipation in nano−CMOS circuits.

Several research results on low-power high-level synthesis (HLS) have been presented in the current literature. Most of the existing research have considered dynamic power reduction without considering process variation [6], [7], [8]. However, if the variations are not estimated properly and existing worst-case analysis is used, leakage power may be exceed the power limit of the design which degrades circuit performance. In [1], authors analyzed the multi−$V_{th}/V_{dd}/T_{ox}$ design space with consideration of process variation at the logic level. A recent research on parametric yield-driven HLS is presented in [9]. Power reduction based on muti−V_{dd}/V_{th} requires extra power supply voltages and is not applicable in performance-critical circuit design. Thus, variation-aware low-power exploration for architecture synthesis to ensure maximum yield needs significant research. The current paper presents the impact of process variations on the muti−V_{dd}/V_{th} techniques at the architecture level. The gate-oxide leakage current (I_{ox}) in nano-CMOS circuits is described using following expression [25]:

$$I_{ox} \propto (V_{dd}/T_{ox})^2 \; exp\,(-\gamma\, T_{ox}/V_{dd}). \qquad \text{----------} \qquad (1)$$

Where γ is an experimentally derived factor. So, I_{ox} is proportional to the square of supply voltage and inversely proportional to the square of gate-oxide thickness (T_{ox}). From which it is evident that reducing supply voltage will increase the delay of the circuit and hence would affect the performance of the design. On the other hand, increase in the gate-oxide thickness leads to increase in propagation delay. So, multiple gate-oxide thickness can serve as a leakage power and delay trade-off. In [10], authors have used dual−T_{ox} based *CMOS* technology to minimize the leakage current during high-

level synthesis. However, they did not consider delay variations of the functional unit and their RTL generation is not optimal. In the current paper, a variation-aware gate-oxide leakage power minimization technique is presented in the context of high-level synthesis.

The rest of the paper is organized as follows. Contributions of the paper are summarized in Section II. Section III provides a general background on the variation-aware high-level synthesis. The Taylor Expansion Diagram (TED) is explained in Section IV. The proposed algorithm for variation-aware Nano−CMOS RTL leakage optimization is presented Section V. Section VI describes an algorithm for latency reduction through variation-aware HLS. Experimental results are presented in Section VII. The paper is concluded in Section VIII.

2 CONTRIBUTION OF THIS PAPER TO THE STATE OF THE ART

The paper presents a variation-aware leakage power optimization approach using multi–T_{ox} assignment in the context of high-level synthesis. In this paper, the Taylor Expansion Diagrams (TEDs) representation for high-level design description is used to generate the optimal RTL at the end of synthesis process [12], [13]. This representation is beneficial for modeling and supporting equivalence verification of designs specified at the behavioral level. TED is a canonical, graph based representation, similar to binary decision diagrams (BDDs) and binary moment diagrams (BMDs) [14], [15]. In contrast to BDDs and BMDs, TED is based on a non-binary decomposition principle and is modeled along the Taylor series expansion. TED is capable of capturing an entire class of structural solutions, rather than a single data flow graph (DFG). TED is converted into a structural representation (i.e. DFG) by using decomposition which is optimized for a particular design objective. After obtaining DFG, statistical timing and power analysis is performed to determine delay and power distribution through DFG. The impact of process variation on propagation delay and leakage power is examined for this purpose. A variation-aware datapath component library is constructed in which the characterization is performed for delay and power distribution for different oxide thickness. A variation-aware simultaneous scheduling and resource binding algorithm is presented which takes time constraint as a performance (or propagation delay) trade-off factor and offers user to enhance leakage power yield. The algorithm schedules nodes of DFG at the appropriate control steps and simultaneously binds them to the best available resources while considering constraints to achieve the desire performance with maximum leakage power yield. **The novel contributions of the paper are summarized as follows**:

- To best of authors' knowledge so far, this is the first research to use TED techniques during high-level synthesis in presence of both delay and leakage power variation.
- The HLS flow for variation-aware leakage power optimization in multi−T_{ox} is introduced.
- Consideration of both resource and time constraints to provide user an optimal RTL by taking account of process variations.
- A heuristic algorithm for latency reduction under a given performance yield.

3 POWER, LEAKAGE, DELAY, AND YIELD TRADE-OFFS AT RTL

This section briefly discusses preliminaries on variation-aware high-level synthesis (HLS) and presents the motivation of the current research.

3.1 Timing and Power Yield in HLS

HLS is a process of translating a behavior description into a register-transfer level (RTL) structural description. Scheduling and resource binding are key steps during the HLS. The scheduler

divides the set of arithmetic and logical operation in the DFG into groups so that the operations in the same group can be executed concurrently, while taking into consideration possible trade-offs between total execution cost and hardware cost. The binding process selects resources from the library, which involves trade-offs according to different features like delay, area, power, and leakage. The resource library contains different functional units with different characteristics such as delay, leakage, etc. Traditional HLS techniques consider worst-case latency of each functional unit during scheduling and binding. However, as the magnitude of process variation grows rapidly, worst-case based analysis and optimization are no longer acceptable since they introduce too much pessimism in the design. This in turn creates problem for designers to meet the requirement. Instead, statistical description and analysis of functional units are needed to tackle the timing problem [16], [17], [18], [19], [20], [21], [22], [27]. A few of this research also addresses the power dissipation problem.

In presence of process variation, the delay of the functional unit is no longer a fixed value, but spreads into statistical distributions. In a statistical timing view, the distribution can be described by a probability density function (PDF). Timing yield is defined as the probability that a functional unit can finish execution in a given time period. Alternatively, it is the cumulative probability under a given T_{clk} in PDF. An example of delay and power variation is shown in Fig. 1. The concept of timing yield is represented in Fig. 2. Given the clock time T_{clk}, the overall timing yield of the DFG is the probability that the datapath can finish execution within T_{clk}, and is defined as follows:

$$Yields = P(t_1 \leq T_{clk}, t_2 \leq T_{clk}, \ldots, t_n \leq T_{clk}), \quad \ldots\ldots (2)$$

where $P()$ is the probability function, $t_1, t_2, \ldots, t_n$ are the execution time for the control steps $1, 2, \ldots, n$, respectively.

Dynamic power in CMOS circuit is relatively less sensitive to process variation and it affects the mean value of the total power dissipation. Thus, the current paper applies statistical analysis to the leakage power. The total leakage power consumption of a DFG is calculated by adding leakage power of all functional units present in the DFG. Given a power limit P_L, the power yield of the DFG ($Yield_P$) is defined as the probability that of total power consumption of DFG (P_{DFG}) is less than or equal to P_L, and can be defined as follows:

$$Yield_P = P(P_{DFG} \leq P_L). \quad \ldots\ldots\ldots\ldots (3)$$

In variation-aware HLS, a metric called parametric yield is introduced in [9]. The parametric yield is defined as the probability of the synthesized hardware meeting a specified constraint $Yield = P(Y \leq Y_{max})$, where Y is either delay (D) or gate leakage (current I_{ox} or power P_{ox}).

An example that compares yield-driven approach and worst-case deterministic approach is shown in Fig. 3. The four functional units F_1, F_2, F_3, and F_4 are of same type (e.g. adder or multiplier). The gate-oxide thickness (T_{ox}) of these functional units increases in the order from F_1 to F_4. The leakage power and delay distribution of these units are shown in Fig. 3. The power limit P_L and clock cycle time T_{clk} are also shown in Fig. 3. As T_{ox} of F_1, F_2, F_3, and F_4 follows $T_{ox}(F_1) < T_{ox}(F_2) < T_{ox}(F_3) < T_{ox}(F_4)$, the mean gate-leakage power follows $\mu_{Pox}(F_4) < \mu_{Pox}(F_3) < \mu_{Pox}(F_2) < \mu_{Pox}(F_1)$ and delay follows up as $\mu_D(F_1) < \mu_D(F_2) < \mu_D(F_3) < \mu_D(F_4)$. In the worst-case deterministic approach, F_4 will be chosen under leakage power constraint as it has lowest leakage power consumption. However, from statistical point of view, F_4 has low timing yield and may cause timing violation. Similarly, in the worst-case deterministic approach, F_1 will be chosen from the performance constraint point of view as it satisfies timing constraint. However, F_1 has larger leakage power and may lead to higher power dissipation. Moreover, if both power and performance constraints are simultaneously considered, then F_2 and F_3 are selected. Selection of F_3 results in some loss in timing yield but satisfies power yield, while F_2 results in some loss in power yield but

satisfy timing yield. So, selection of F_2 and F_3 introduces a concept of tradeoff in between timing and power yield. Thus, a yield-driven statistical approach is needed which selects the functional units so that one parameter yield can be maximized under other parametric yield constraint.

3.2 Library setup for yield-driven multi -T_{ox} optimization

Gate-leakage power is inversely proportional to the gate-oxide thickness (T_{ox}). The increase in T_{ox} results decrease in gate-leakage power and increase in propagation delay. The current paper presents a datapath-component library with different T_{ox} components. For this purpose, a datapath library of 16−bit components, such as adders, subtractors, multipliers, comparators, multiplexers, and registers is characterized using the standard structural descriptions [26]. The datapath library is simulated using different gate-oxide thicknesses. The Predictive Technology Model (PTM) of 45nm technology node is used in this paper, with base values of T_{ox} = 1.4nm , V_{dd} = 0.7V and V_{th} = 0.22V. The effect of varying oxide thickness was incorporated by varying the parameter t_{oxe} in the SPICE deck. It may be noted that the length of the device is proportionately changed to maintain a constant (L/T_{ox}) ratio in order to minimize the impact of higher oxide thickness on device performance and to maintain the per width gate capacitance constant as per fabrication requirements [23]. The PMOS transistors are sized appropriately to ensure proper functionality of the building blocks. The process variation effects are exhaustively evaluated through detailed 10000 Monte Carlo simulations to capture the effects. The primary goal of this analysis is to assess the extent of gate-leakage I_{ox} and power variation as a result of process variations in gate oxide thickness T_{ox}. The distribution of the device parameters is assumed to be Gaussian with the variance of 10%. The statistical variation of the delay corresponding to oxide thickness 1.4nm and 1.7nm is shown in Table I.

Table I indicates the propagation delay values of the datapath components or functional units under different performance yields. In order to obtain these values, the delay distributions for each of the functional units are generated. The delay for a specific timing yield is calculated by finding the area under the curve. For example, Fig. 4 shows the PDF for the adder of Table I. Under 100% yield, the delay of the adder is 11.68 ns (T_{ox} = 1.4 nm) which corresponds to point A in Fig. 4. However, if 10% yield is sacrificed then the delay becomes 10.94 ns which corresponds to point B. Similarly, points C and D represent the delay value of 11.09 ns and 10.98 ns correspond to 97% and 94% of timing yield. Once characterized, the next task is to create the DFG from the behavioral description of the given circuit. In order to create optimized DFG, in the present work we have used TED based approaches which is described in the next section.

4 CANONICAL TED FOR HIGH-LEVEL REPRESENTATION: A FINITE IMPULSE FILTER (FIR) CASE STUDY

Taylor Expansion Diagram (TED) is a canonical, word-level data structure that offers an efficient way to represent computation in a compact, factored form. An Algebraic, multi-variable expression $f(x, y, ..)$, is represented using Taylor series expansion for variable x as follows:

$$f(x, y, \ldots) = f(x = 0) + x\, f'(x = 0) + (1/2)\, x^2 f''(x = 0) + \ldots \quad . \qquad (4)$$

Where $f'(x)$, $f''(x)$, … , are the successive derivatives of function f with respect to x. The terms of the decomposition are then decomposed with respect to the remaining variables (y, …, etc.), one variable at a time. A directed acyclic graph is used to store the resulting decomposition whose nodes represent the terms of the expansion. The detailed explanation of TED can be found in [12], [13].

This paper considers digital signal processing (DSP) application with finite-impulse response (FIR) filters as specific example circuits. The polynomial corresponding to a 4−tap FIR filter can be written as follows:

$$Y_n = a_0 X_n + a_1 X_{n-1} + a_2 X_{n-2} + a_3 X_{n-3}. \quad (5)$$

The TED corresponding to Eqn. 5 is shown in Fig. 5. Given an optimized TED, the next task is to convert it to DFG as shown in Fig. 6. An STA on DFG is performed to generate the necessary timing information. Specifically, calculation is needed for the arrival time T_a, required time T_r, and slack $T_s = T_r - T_a$, for each node in DFG.

Definition 1: Arrival time T_a of a *DFG* node *n* is recursively defined as a sum of delay of node *n* and the maximum arrival time of its inputs:

$$T_a(n) = Delay(n) + max(T_a(n_i)|_{n^i \in Input(n)}), \quad (6)$$

where *Delay*(*n*) denotes the delay of the operation associated with node *n*, and *Input*(*n*) is the set of input nodes to the node *n*.

Definition 2: Required time T_r of a node *n* is recursively defined as a difference between the minimum required time of its outputs and delay of node *n*:

$$T_r(n) = min(T_r(n_o)|_{no \in Output(n)}) - Delay(n), \quad (7)$$

where, *Output*(*n*) is the set of output *DFG* nodes of node *n*.

Definition 3: Slack time T_s of a *DFG* node *n* is defined as a difference between its required time T_r and the arrival time T_a which is expressed as follows:

$$T_s(n) = T_r(n) - T_a(n). \quad (8)$$

In Fig. 6, the arrival time T_a, the required time T_r, and the slack T_s of each node are denoted in the form of $[T_a/T_r/T_s]$; where for simplicity, it is assumes that the delay of each functional unit is 1. Based on the definition of slack, a critical node and critical path in DFG can be identified as follows.

Definition 4: A critical node in a DFG is a node which has a slack equal to 0. A critical path is a path which contains critical nodes only.

In Fig. 6, critical path 1 consists of 4 nodes (*M1, A1, A2, A3*) and critical path 2 consists of 4 nodes (*M2, A1, A2, A3*). In the next subsection, the generalized algorithm is presented for variation-aware simultaneous scheduling-binding for general circuits.

5 AN ALGORITHM FOR VARIATION-AWARE NANO− CMOS RTL LEAKAGE OPTIMIZATION

This section presents a simultaneous scheduling and binding algorithm under resource constraint. The inputs to the algorithm are an unscheduled DFG, libraries with different resources made of transistors of different gate-oxide thickness, a delay trade-off factor T_d, and performance yield Y_d. The T_d is a user defined quantity which specifies the maximum allowed critical path delay of the target circuit. The present algorithm schedules the DFG in such a way that critical path delay is either less than or equal to T_d while improves the power yield under performance yield constraint Y_d.

The proposed algorithm performs an initial STA on the DFG to identify critical and non-critical nodes by calculating T_a, T_r, and T_s of each nodes. During this step, it uses delay value of 1 for each node. Once identified, it assigns T_{oxL} (low gate-oxide component) to critical nodes and T_{oxH} (high gate-oxide components) to non-critical nodes. After initial scheduling and binding, it calculates critical path delay and power yield. Now the algorithm searches iteratively on the DFG to reduce the leakage power or improves the power yield under Y_d and T_d constraints. Or in other words, it replaces

nodes of T_{oxL} with T_{oxH} so that power yield can be maximized by satisfying Y_d and T_d. The pseudo code of the algorithm is presented in Algorithm 1.

Consider the FIR filter of Fig. 5 under the assumption that unlimited number of T_{oxL} and T_{oxH} components. The T_d and Y_d are assumed to 10 ns and 90%, respectively. The delay of the library units under 100% and 90% timing yield is shown in Table II for both T_{oxL} and T_{oxH} components.

The DFG of Eqn. 6 is shown in Fig. 7 after initial scheduling and binding where critical nodes are bound to T_{oxL} and noncritical nodes to T_{oxH}. During this phase, the algorithm uses delay values of the functional units corresponding to 100% timing yield (or worst-case analysis). Once scheduled, the algorithm searches iteratively to bind T_{oxL} node with T_{oxH} components under Y_d and T_d constraints. It is clear from Fig. 7 that *M1* can be replaced with T_{oxH} as under 90% yield delay of the multiplier is 4ns (see Table II). After replacement, timing yield of the DFG will be 90% which is equal to Y_d. Similarly, instead of *M1* one can also replace *A3* with T_{oxH} components. However, the replacement of multipliers saves much more leakage power than an adder. The scheduled DFG is shown in Fig. 8.

6 LATENCY REDUCTION THROUGH VARIATION-AWARE HLS

This section presents the principle behind the latency reduction through variation-aware HLS under timing and power yield constraints. An algorithm to reduce the latency is also proposed.

6.1 Preliminary

This section describes the latency minimization in the framework of HLS under timing and power yields constraints. Fig. 9 shows an instance of scheduling and binding of the FIR filter of Fig. 6 produced by variation-unaware HLS. It produces latency of 6 clock cycles by using the worst case module delay. However, parametric yield driven HLS is able to perform more aggressive scheduling and binding. As shown in Fig. 10, the same data flow graph can be schedule at 5 clock cycles by scheduling two operations (op1, op2) and (op3, op4) at two clock steps. Thus, one clock cycle is saved at the expense of some performance yield loss. The objective can be stated as follows: Given a performance yield, schedule and bind the DFG so that latency and power yield are minimized.

However, important issue is how to calculate the performance and power yield of the scheduling and binding instances in Fig. 10, using reliable delay and power distribution of the module. In the current paper, it is assumed that distributions of different modules are independent of each other. Let $D1(=sum(D_{mult1},D_{add}))$ and $D2(= sum(D_{mult2},D_{add}))$ be the delay variable of the two combined operations (*op1, op2*) and (*op3, op4*), respectively, where D_{mult1}, D_{mult2}, and D_{add} are the delay distribution for the modules *multiplier 1* (*Mult1*), *multiplier 2* (*Mult2*), and *adder* (*Add*), respectively. The performance yield of the scheduling and binding process of Fig. 10 can be expressed as Y_d ($D_1 \leq 2t_{clk}$,$D_2 \leq 2t_{clk}$) and is calculated as follows:

$$Y_d = \int_0^{2tclk} (D_{add}(t) * \int_0^{2tclk - t} (D_{mult1}(s)\, ds)\, dt), \qquad (9)$$

where t is the time required for the adder. In other word, when adder needs time t (which varies from *0* to $2t_{clk}$) to finish execution, *Mult1* could have maximum execution time of ($2t_{clk} - t$), in order not to violate the timing. In the next subsection, a variation-aware scheduling and resource binding algorithm is proposed to minimize the latency and leakage power yield under a specific timing yield constraint.

6.2 Variation-aware latency reduction algorithm

The proposed algorithm iteratively searches on a scheduled DFG to reduce the latency under a performance yield constraints. Initially, it schedules and binds the DFG as in [24] in order to achieve 100% performance yield and leakage power yield. Then next step finds number of possible moves of the operations among the control steps (clock cycles). For each movement, it calculates the gain in the latency and stores it in the gain list. Once the gain list is populated, the optimal sequence of moves is then applied to the DFG. For each move, it calculates the performance yield. If the performance yield is within the accepted value, it allows that move otherwise not. The process continues until all the moves have been checked or the performance yield has reached the value set by the user. The gain in each move is calculated in terms of clock cycles. Suppose, the delay distribution of the multiplier module provides the information that 94% of the operation can complete in 3 clock cycles and the remaining 6% needs 4 clock cycles to complete the operation. If the performance yield constraint is set to 90% we can schedule the multiplier in 3 clock cycles which results a gain of 1 clock cycle. The overall HLS flow of latency minimization under performance yield constraint is shown in Fig. 11. The steps are outlined in Algorithm 2 [24].

7 EXPERIMENTAL RESULTS

This section presents the experimental results of the variation-aware leakage power yield improvement framework for HLS. The present algorithms are implemented in C and tested on several high-level benchmark circuits with several constraints [25]. The resource constraints are expressed as the functional units of different oxide thickness, time constraints in terms of performance trade-off factor (T_d), and time yield in terms of performance yield factor (Y_d). The experimental results are presented for selected set of benchmark circuits and constraints.

The comparison of variation-aware leakage optimization algorithm against traditional deterministic (worst-case) approach is shown in Table III. The dual oxide thickness pair of 1.4nm−1.7nm is used with 1.4nm as the nominal thickness. Different resource constraints are used to show the effect of oxide thickness on performance yield and gate-oxide leakage current (I_{ox}). In first case, a smaller number of T_{oxH} resources and a higher number of T_{oxL} are used. This case is designated by 1 in the 2nd column of Table III. In the second case, a higher number of T_{oxH} resources are used as compared to the first set of experiment, and so on. In final case, the resource constraints are relaxed to study the time and performance yield constraints only. This case is designated by the ∞ in the 2nd column of Table III. In each case, I_{ox} has been calculated under a different T_d and Y_d constraints. The present algorithm is then compared against the traditional deterministic approach, which uses worst-case delay value of the functional units in the multi−T_{ox} library. The worst-case based approach leads to 100% timing yield. This corresponds to Y_d = 100% in Table III. However, this dissipates high I_{ox}. But, when relax Y_d to 95% (or 90%), a lower I_{ox} is resulted which indicates improvement to the leakage current. The percentage improvement of I_{ox} against worst-case approach is also reported in Table III and indicates by ΔI. The results indicate significant reduction in I_{ox} when we scarify some of performance yield. The CPU execution time for the algorithms is in the range of 20 sec to 60 sec as reported in Table IV.

The leakage power yield improvement against worst-case delay based approach under timing yield constraints 95% and 90% for different benchmark circuits when T_d = 1.2ns is shown in Fig. 12. The experimental results indicate that the power yield improvement depends on how much timing yield loss is affordable for the design.

The results produced by the proposed variation-aware latency reduction algorithm under different performance yields is shown in Table IV. The latency reduction in the table represents the

reduction of the latency over the traditional deterministic approach. The power yield is also presented in the Table IV.

Since there is no high-level synthesis research dealing with gate-oxide leakage reduction under different performance yield, direct comparison is not possible. However, in view of power yield improvement a broader comparative perceptive is presented in Table V with [9]. Δp indicates the power yield improvement against deterministic worst-case approach under different performance yield. From Table V, it is clear that multi$-T_{ox}$ is an attractive approach for improving performance yield at the same time reducing gate leakage current for nano−CMOS datapath circuits.

8 CONCLUSIONS

This paper presented an effective way to solve the problem of variation-aware HLS. The timing and leakage power constraints based scheduling and resource binding algorithm is introduced for statistical HLS. The TED based approaches is used to generate the optimize DFG. The proposed algorithm maximizes the leakage power yield of the design circuit for a given performance constraint. The experimental results on several benchmark circuits show that performance yield can be maintained with increasing leakage power yield. The future research in this direction will include carbon nanotube FET (CNTFET) technology based circuit optimization during RTL synthesis.

9 ACKNOWLEDGMENTS

This research was partially funded by the Engineering and Physical Science Research Council (EPSRC), UK, under Grant No: EP/G032904/1 and EP/F030991/1. Author S. P. Mohanty acknowledges NSF awards CNS−0854182 and DUE-0942629, for partial support for this research. This archival journal paper is based on the previous shorter-version conference publication [11].

REFERENCES

[1] A. Srivastava, D. Sylvester, and D. Blaauw, "Statistical Analysis and Optimization for VLSI: Timing and Power", Springer, 2005.

[2] A. Agarwal, B. C. Paul, H. Mahmoodi, A. Datta, and K. Roy, "A process-tolerant cache architecture for improved yield in nanoscale technologies", *IEEE Transactions on VLSI Systems*, Dec. 2005, vol. 13, pp. 27 – 38.

[3] Y. Xie and Y. Chen, "Statistical High-Level Synthesis under Process Variability", *IEEE Design & Test of Computer*s, 2009, vol. 26, pp. 78–87.

[4] K. Kuhn, et al., "Managing Process Variation in Intel's 45nm CMOS Technology", *Intel Technology Journ*al, Vol. 12, Issue 02, June 2008.

[5] S. Borkar, T. Karnik, S. Narendra, J. Tschanz, A. Keshavarzi, and V. De, "Parameter variations and impact on circuits and microarchitecture", in *Proceedings of the Design Automation Conference*, 2003, pp. 338- 42.

[6] K. S. Khouri and N. K. Jha, "Leakage power analysis and reduction during behavioral synthesis", *IEEE Transactions on VLSI Systems*, Dec. 2002, vol. 10, pp. 876 – 885.

[7] Wen-Tsong Shiue, "High level synthesis for peak power minimization using ILP", in *Proceedings of the IEEE International Conference on Application-Specific Systems, Architectures, and Processors*, 2002, pp. 103 – 112.

[8] K. Usami and M. Igarashi, "Low-power design methodology and applications utilizing dual supply voltages," in *Proceedings of the Asia and South Pacific Design Automation Conference*, 2000, pp. 123 – 128.

[9] Y. Chen, Y. Xie, Yu Wang, and A. Takach, "Parametric yield driven resource binding in behavioral synthesis with multi-Vth/Vdd library", in *Proceedings of the 15th Asia and South Pacific Design Automation Conference*, 2010, pp. 781 – 786.

[10] S.P. Mohanty, E. Kougianos, and D. K. Pradhan, "Simultaneous scheduling and binding for low gate leakage nano-complementary metal-oxide-semiconductor data path circuit behavioural synthesis", *IET Computers and Digital Techniques*, Volume 2, Issue 2, March 2008, pp. 118 – 131.

[11] S. Banerjee, J. Mathew, D. K. Pradhan, S. P. Mohanty, and M. Ciesielski, "Variation-Aware TED−Based Approach for Nano−CMOS RTL Leakage Optimization," in *Proceedings of the International Conference on VLSI Design*, pp. 304--309, 2011.

[12] M Ciesielski, J. Guillot, D. Gomez-Prado, E. Boutillon, "High-Level Dataflow Transformations Using Taylor Expansion Diagrams," *IEEE Design and Test of Computers*, 2009, vol. 26, Issue 4, pp. 46 – 57.

[13] M. Ciesielski, P. Kalla, and S. Askar, "Taylor Expansion Diagrams: A Canonical Representation for Verification of Data Flow Designs," *IEEE Transactions on Computers*, Sep 2006, vol. 55, pp. 1188 – 1201.

[14] R. E. Bryant, "Graph-Based Algorithms for Boolean Function Manipulation," *IEEE Transactions on Computers*, Aug 1986, vol. 35, pp 677 – 691.

[15] R.E. Bryant and Yirng-An Chen, "Verification of Arithmetic Circuits with Binary Moment Diagrams," in *Proceedings of 32nd Conference on Design Automation*, 1995, pp. 535 – 541.

[16] Feng Wang, Guangyu Sun and Yuan Xie, "A Variation Aware High Level Synthesis Framework," in *Proceedings of the Design, Automation and Test in Europe*, 2008, pp. 1063 – 1068.

[17] J. Jung and T. Kim, "Timing variation-aware high-level synthesis," in *Proceedings of the IEEE/ACM International Conference on Computer-Aided Design*, Nov 2007, pp. 424 – 428.

[18] J. Jung and T. Kim, "Timing variation-aware high level synthesis: Current results and research challenges," in *Proceedings of the IEEE Asia Pacific Conference on Circuits and Systems*, 2008, pp. 1004 – 1007.

[19] Yibo Chen and Yuan Xie, "Tolerating process variations in high-level synthesis using transparent latches", in *Proceedings of the Asia and South Pacific Design Automation Conference*, 2009, pp. 73 – 78.

[20] W. -L. Hung, Xiaoxia Wu, and Yuan Xie, "Guaranteeing Performance Yield in High-Level Synthesis," in *Proceedings of the IEEE/ACM International Conference on Computer-Aided Design*, 2006, pp. 303–309.

[21] A. Muttreja, S. Ravi, and N. K. Jha, "Variability-Tolerant Register-Transfer Level Synthesis", in *Proceedings of the IEEE International Conference on VLSI Design*, 2008, pp. 621–628.

[22] S. P. Mohanty, and E. Kougianos, "Simultaneous Power Fluctuation and Average Power Minimization during Nano-CMOS Behavioral Synthesis," in *Proceedings of the 20th IEEE International Conference on VLSI Design*, 2007, pp. 577–582.

[23] Y. Taur, "CMOS Design Near the Limits of Scaling," *IBM Journal on Research and Development*, Volume 46, Issue 2-3 , Mar/May 2002, pp. 131 – 139.

[24] S. Banerjee, J. Mathew, D. K. Pradhan, S. P. Mohanty, and M. Ciesielski, "A Taylor Expansion Diagram Approach for Nano-CMOS RTL Leakage Optimization," in *Proceedings of the International Symposium on Electronic System Design (ISED)*, pp. 71—76, 2010.

[25] S. P. Mohanty, N. Ranganathan, E. Kougianos, and P. Patra, "Low-Power High-Level Synthesis for Nanoscale CMOS Circuits", Springer, 2008.

[26] N. H. E. Weste and D. Haris, "CMOS VLSI Design: A Circuits and Systems Perspective," Addison Wesley, 2005.

[27] A. Davoodi and A. Srivastava, “Power-Driven Simultaneous Resource Binding and Floorplanning: A Probabilistic Approach”, *IEEE Transaction on VLSI Systems*, Vol. 13, No. 8, pp. 934-942, August 2005.

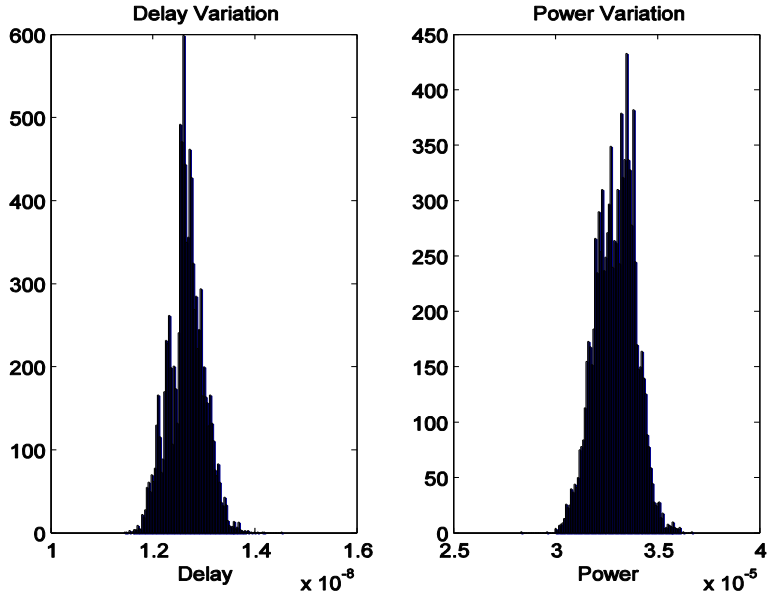


Figure 1. The delay and power variations of an adder implemented in 45nm technology.

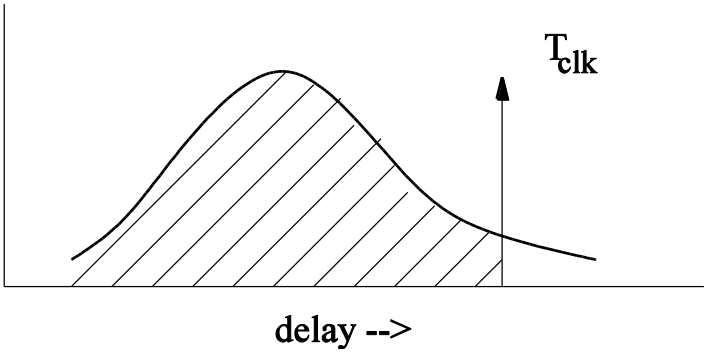


Figure 2. Timing yield of an adder with clock period T_{clk}.

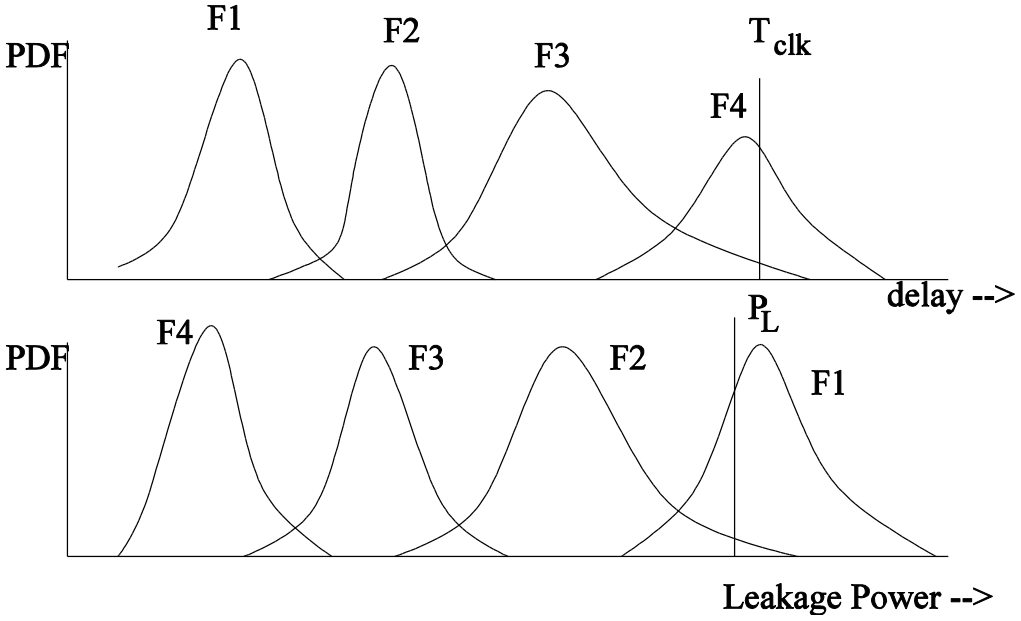


Figure 3. Delay and leakage power distributions of the functional unit. (PDF represents the probability density function).

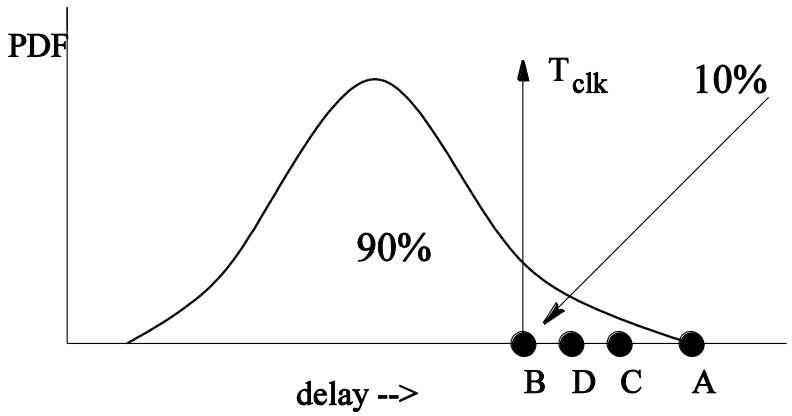


Figure 4. The delay values of an adder under different timing yield. 10% represents the area under the curve between points B and A while 90% indicates area between starting point and B.

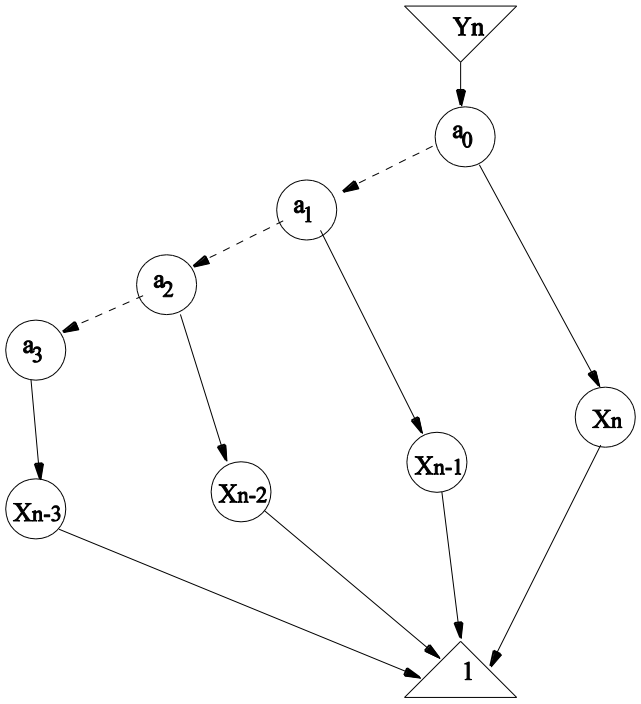


Figure 5. TED for a 4−tap FIR filter.

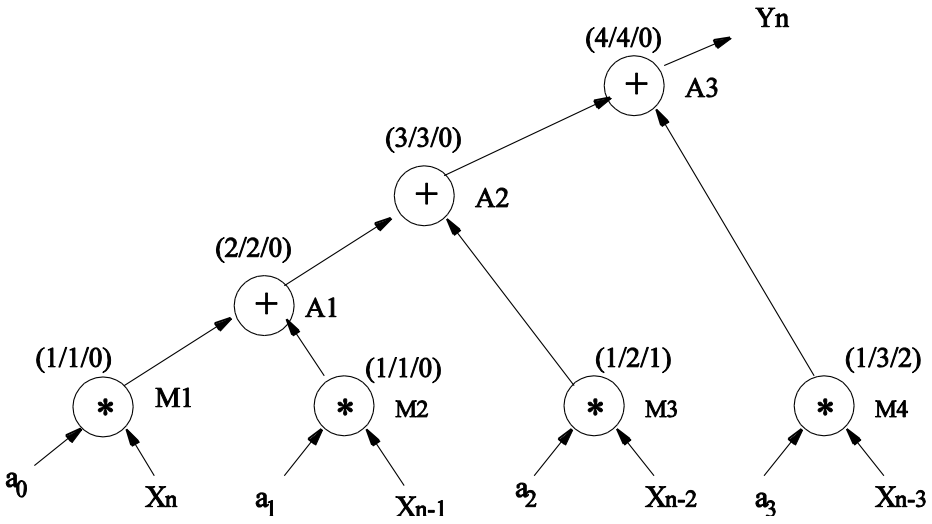


Figure 6. DFG for the TED of Fig. 5.

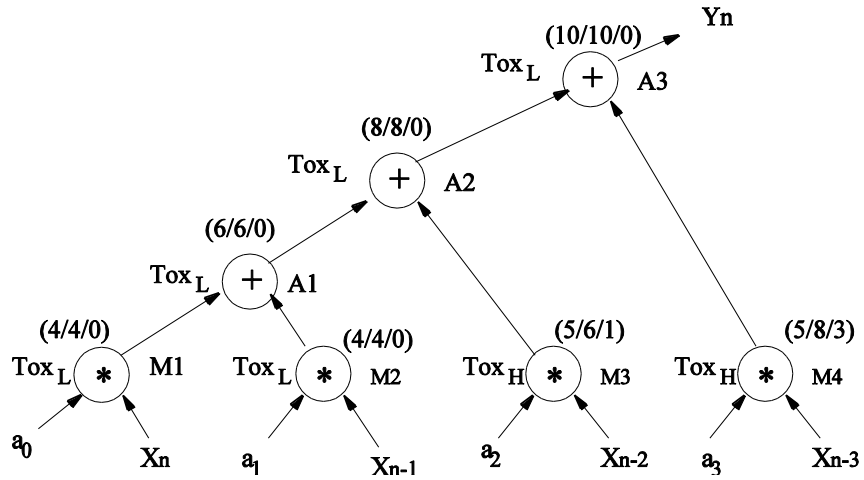


Figure 7. The DFG of Fig. 5 after initial scheduling and binding.

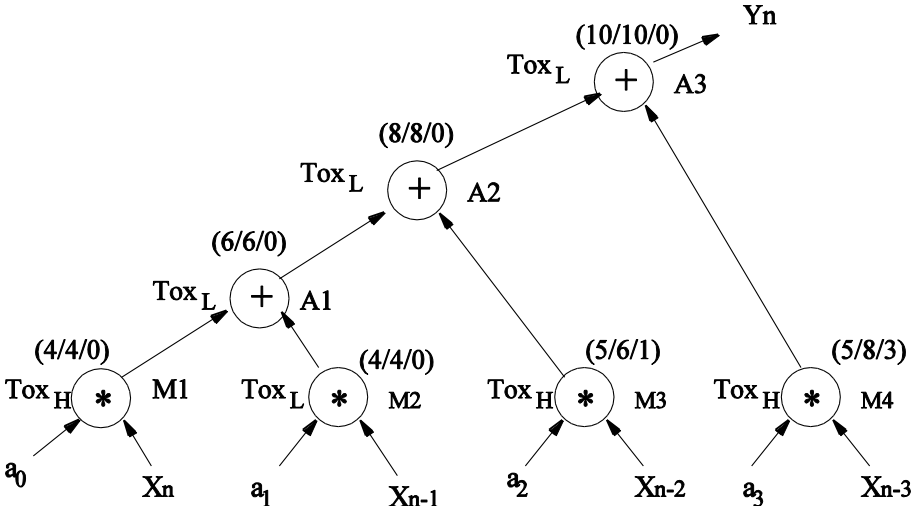
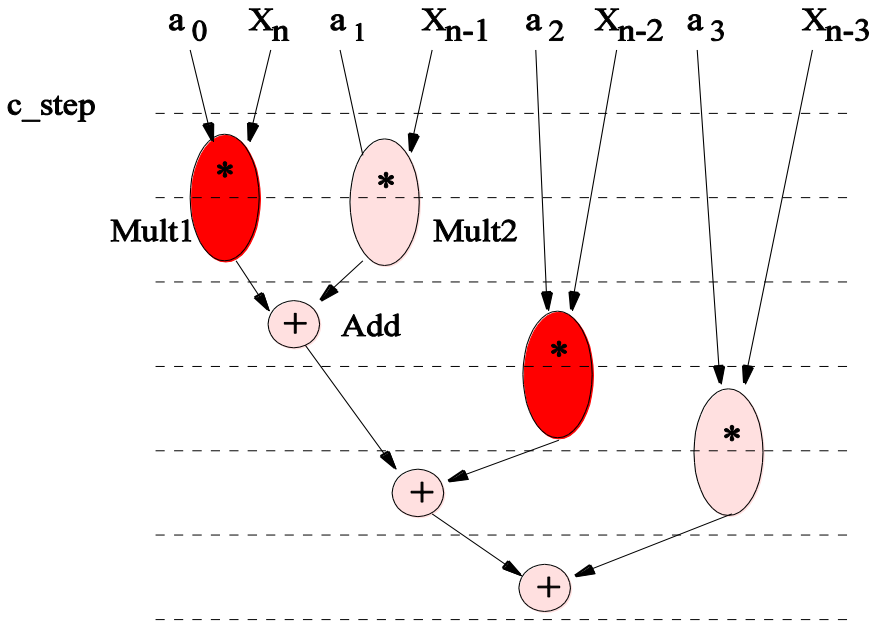
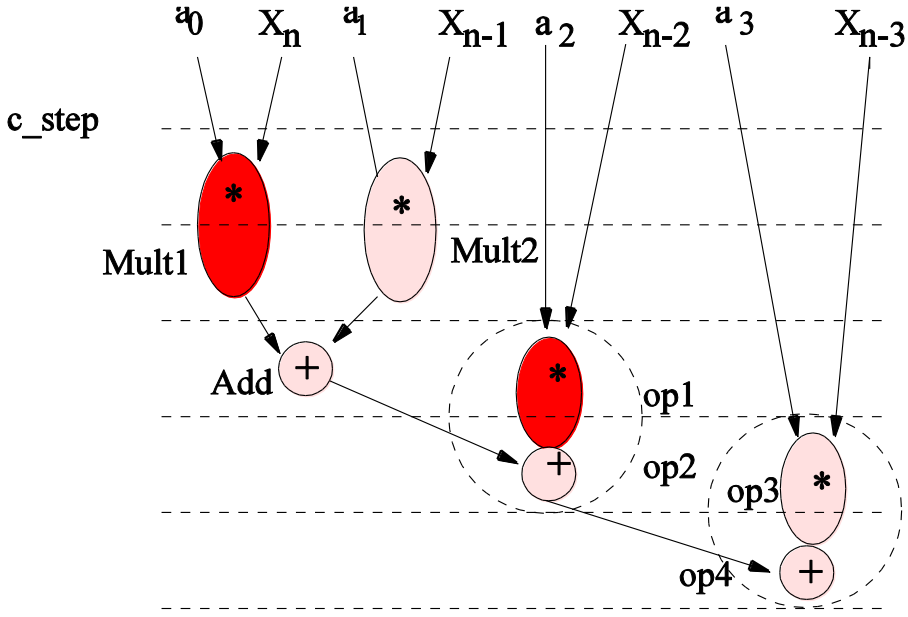


Figure 8. Final *DFG* after variation-aware scheduling and binding.



timing yield = 100%, power yield = 100%, latency = 6

Figure 9. Process variation-unaware scheduling and binding results using worst case delays.



timing yield = 86%, power yield = 95%, latency = 5

Figure 10. Process variation-aware scheduling and binding results using statistical delays.

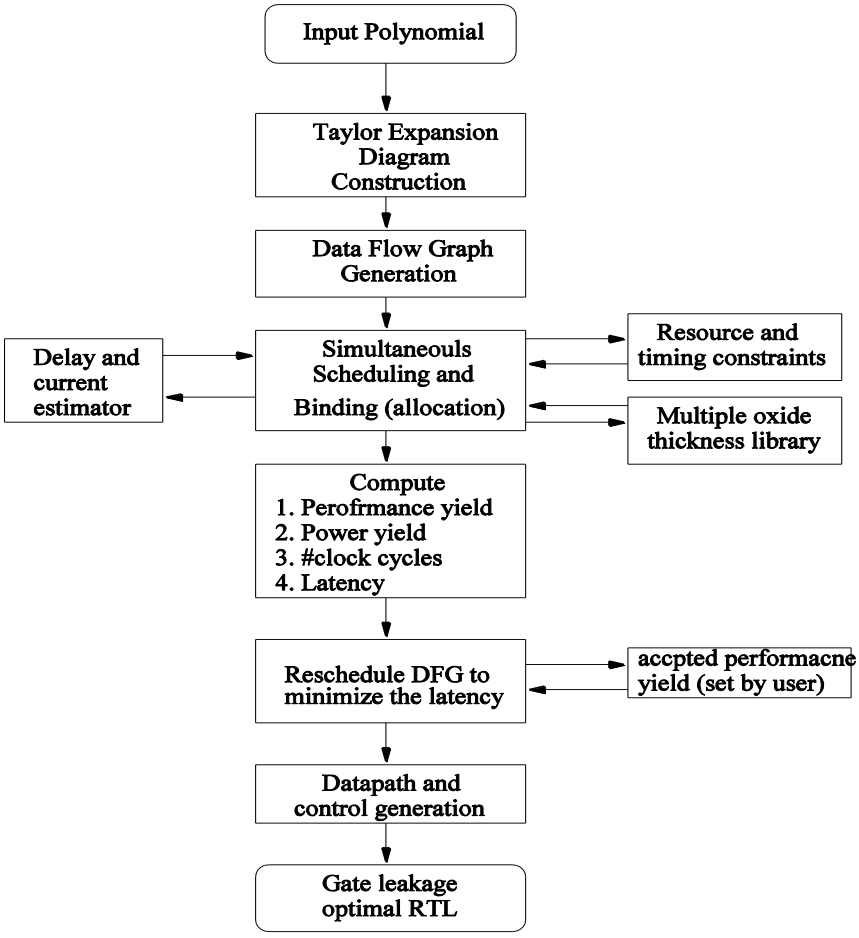


Figure 11. The high-level synthesis framework for latency reduction.

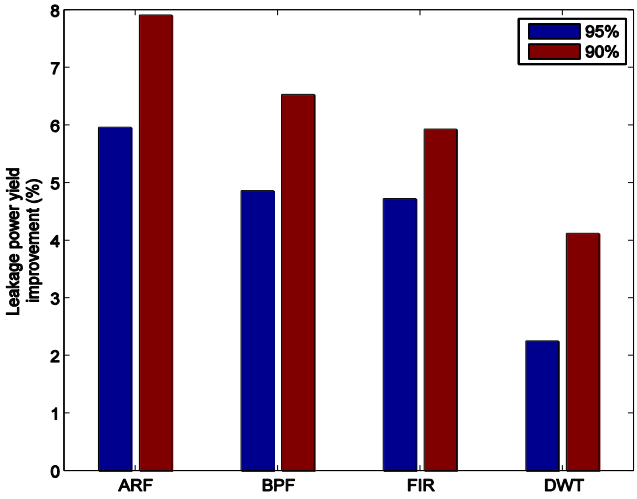


Figure 12. Leakage power yield improvement against worst-case approach (95% and 90% in the figure represent timing yield).

TABLE I
LIBRARY WITH DIFFERENT GATE-OXIDE THICKNESS

Functional unit	T_{ox} = 1.4nm					T_{ox} = 1.7nm				
	I_{ox} (μA)	T_{pd} (ns)				I_{ox} (μA)	T_{pd} (ns)			
		Yield 100%	Yield 97%	Yield 94%	Yield 90%		Yield 100%	Yield 97%	Yield 94%	Yield 90%
Adder	2.155	11.68	11.09	10.98	10.94	0.2725	14.52	13.30	13.20	13.12
Subtractor	11.99	11.46	10.17	10.17	10.16	3.185	14.59	13.33	13.23	13.15
Multiplier	53.81	15.55	15.48	15.45	15.44	6.701	17.29	16.55	16.49	16.45
Comparator	3.30	0.2304	0.2274	0.2269	0.2266	0.123	0.2396	0.2309	0.2308	0.2307
Register	3.465	0.7934	0.7732	0.7662	0.7599	0.2025	0.7973	0.7646	0.7586	0.7534
Multiplexer	3.181	0.3763	0.3738	0.3735	0.3732	1.827	0.3997	0.3833	0.3832	0.3830

TABLE II
DELAY VALUES UNDER DIFFERENT TIMING YIELD

Functional Unit	Delay (ns) (for T_{oxL})		Delay (ns) (for T_{oxH})	
	Yield 100%	Yield 90%	Yield 100%	Yield 90%
Adder	2	1	3	2
Multiplier	4	3	5	4

TABLE III
EXPERIMENTAL RESULTS FOR THE PROPOSED ALGORITHM

Benchmarks	Resource Constraints	T_d = 1.0 ns					T_d = 1.2 ns				
		Y_d					Y_d				
		100%	95%		90%		100%	95%		90%	
		I_{ox} (μA)	I_{ox} (μA)	ΔI	I_{ox} (μA)	ΔI	I_{ox} (μA)	I_{ox} (μA)	ΔI	I_{ox} (μA)	ΔI
ARF	1	886.8	865.1	2.4	854.7	3.7	853.4	847.1	0.0	839.4	1.6
	2	860.4	840.6	2.3	828.1	3.8	839.9	822.2	2.0	811.1	3.4
	3	835.1	810.4	3.0	798.1	4.6	810.6	796.8	1.7	780.1	3.8
	∞	790.6	767.5	2.9	752.7	5.0	774.1	749.1	3.3	730.4	5.4
BPF	1	667.5	652.5	2.3	645.6	3.4	654.1	643.9	1.7	635.7	2.9
	2	645.1	633.1	1.8	624.5	3.3	636.8	621.1	2.4	615.4	3.4
	3	631.6	621.9	1.6	607.1	3.9	624.1	607.7	2.4	599.1	3.1
	∞	606.6	594.1	2.0	586.2	4.0	595.3	582.4	2.8	578.9	3.9
FIR	1	462.8	451.5	2.4	445.6	3.8	449.7	438.5	2.5	431.9	4.1
	2	453.4	441.1	2.7	436.7	3.8	435.6	426.1	2.1	418.3	4.1
	3	446.7	434.6	2.7	423.3	5.4	427.1	415.9	2.8	408.0	4.6
	∞	414.5	401.4	3.2	390.8	6.1	398.5	391.3	1.7	383.8	3.9
EWF	1	486.5	478.4	1.6	470.9	3.4	471.2	462.6	1.9	456.2	3.2
	2	470.2	461.8	1.9	453.1	3.7	458.1	447.3	2.4	438.1	4.5
	3	462.8	452.1	2.2	440.5	5.0	443.8	432.8	2.5	421.6	5.2
	∞	441.3	430.5	2.5	418.5	5.5	401.4	401.4	2.9	388.1	5.5
DCT	1	393.5	388.7	1.2	385.6	2.0	382.1	377.5	1.2	371.4	2.8
	2	350.4	343.5	1.9	341.4	2.4	332.9	327.8	1.5	316.6	3.2
	3	342.8	334.0	2.3	331.5	3.4	316.4	309.4	2.2	298.6	3.6
	∞	310.2	301.2	2.6	298.5	3.9	298.7	290.7	2.4	278.2	4.2

TABLE IV
LATENCY REDUCTION FOR THE PRESENT ALGORITHM

Benchmarks	Y_d = 95%			Y_d = 90%		
	Latency Reduction (%)	Power Yield	CPU Time (sec)	Latency Reduction (%)	Power Yield	CPU Time (sec)
ARF	12.2	94.7	20	20.2	93.9	35
BPF	15.0	95.1	30	19.7	94.1	41
FIR	13.8	95.3	35	19.1	95.4	52
EWF	10.5	96.2	40	14.3	94.3	62
DCT	12.8	97.1	38	15.9	96.1	48
HAL	11.2	98.3	28	12.9	97.8	39

TABLE V
A BROAD COMPARATIVE PERCEPTIVE WITH [9]

Benchmarks	Y_d = 95%		Y_d = 90%	
	[9] (Multi−V_{dd}/V_{th}) **Δp**	Current Paper (Multi−T_{ox}) **Δp**	[9] (Multi−V_{dd}/V_{th}) **Δp**	Current Paper (Multi−T_{ox}) **Δp**
EWF	3.9	4.2	4.0	4.4
FIR	4.8	4.6	5.4	5.8
BPF	4.5	4.7	6.6	6.4

Algorithm 1 Nano−CMOS RTL Optimization for Yield, Gate-Leakage, and Time Tradeoff

Apply STA to DFG under resource constraint.
Assume each node is assign to a delay of 1.
Identified critical and non-critical nodes.
for all critical nodes ni **do**
    if *Functional_Unit*$_j$(k, T_{oxL}) is available for control step $C[n_i]$ **then**
        Assign *Functional_Unit*$_j$(k, T_{oxL}) to node n_i.
    else
        Assign *Functional_Unit*$_j$(k, T_{oxH}) to node n_i.
    end if
end for
for all non-critical nodes n_i from root of the *DFG* **do**
    for all possible control steps (slack) of n_i **do**
        if *Functional_Unit*$_j$(k, T_{oxH}) is available for control step $C[n_i]$ **then**
            Schedule n_i in control step $C[n_i]$.
            Assign *Functional_Unit*$_j$(k, T_{oxH}) to node n_i.
            Update T_s for all the nodes connected to n_i.
        end if
    end for
    if n_i is not scheduled **then**
        for all possible control steps (slack) of n_i **do**
            if *Functional_Unit*$_j$(k, T_{oxL}) is available for control step $C[n_i]$ **then**
                schedule n_i in control step $C[n_i]$
                Assign *Functional_Unit*$_j$(k, T_{oxL}) to node n_i
                Update T_s for all the nodes connected to n_i.
            end if
        end for
    end if
end for
Calculate T_a, T_r, and T_s for all nodes.
Calculate critical path delay T_{cp} and power yield Y_p of the DFG.
Sort all critical nodes according to ascending order of leakage current.
Calculate timing yield Y_t of the DFG.
for all critical nodes n_i **do**
    if (T_d greater than T_{cp}) and (Y_t greater than Y_d) **then**
        if *Functional_Unit*$_j$(k, T_{oxH}) is available for control step $C[n_i]$ **then**
            Assign *Functional_Unit*$_j$(k, T_{oxH}) to node n_i.
            Calculate Y_t and modified power yield Y_{pt} of DFG.
                if ((Y_{pt} - Y_p) greater than 0) and (Y_t greater than Y_d) **then**
                    Calculate T_{cp} of the DFG.
                    if T_{cp} less than or equal to T_d **then**
                        $Y_p = Y_{pt}$.
                        Continue to next critical node.
                    end if
                end if
                Assign *Functional_Unit*$_j$(k, T_{oxL}) to node n_i.
        end if
    end if
end for

Algorithm 2 Nano−CMOS latency minimization

1: Schedule and bind the *DFG* using for low leakage power.
2: *Generate_gain_list* G.
3: Sort G based on gains.
4: Calculate the performance yield Y_{td}.
5: Set the performance yield constraints Y_d.
6: **while** *((Y_d - Y_{td})* greater than *0*) and (*G* is not empty)) **do**
7: g_i = *DEQUEUE(G);*
8: Reschedule *DFG* according to g_i.
9: Calculate the performance yield Y_{td}.
10: **if** (*(Y_d - Y_{td})* is greater than *0*) **then**
11: Restore *DFG* as before applying g_i.
12: Calculate the performance yield Y_{td}.
13: Continue
14: **end if**
15: **end while**
16: Evaluate latency and power yield.
17: *Generate_gain_list*.
18: Find all possible moves using heuristics [9].
19: **for all** moves **do**
20: Calculate the gain g_i of the move.
21: **if** gain g_i greater than *0* **then**
22: *G.QUEUE(g_i)*
23: **end if**
24: **end for**
25: return *G*.

BIOGRAPHIES

Shibaji Banerjee received B.Tech and M.Tech degree in Radio physics and Electronics, Calcutta University. He obtained his Ph.D. degree from IIT, Kharagpur, India. Presently, he is doing post doctoral research at University of Bristol. His current research includes approaches for low-power design and test. He is an author of 25 peer-reviewed publications in this area of research.

Jimson Mathew received the Ph.D. degree in computer science from University of Bristol, Bristol, U.K. He has held positions with the Centre for Wireless Communications, National University of Singapore, Bell Laboratories Research (Lucent Technologies), North Ryde, Australia and Royal Institute of Technology (KTH), Stockholm, Sweden. Since 2005, he has been with Department of Computer Science, University of Bristol. His research interest primarily focuses on low power design and testing, sigma delta converters, fault-tolerant computing, and Galois field-based arithmetic.

Saraju P. Mohanty is currently an Associate Professor at the Dept. of Computer Science and Engineering, University of North Texas and the director of the NanoSystem Design Laboratory (NSDL, http://nsdl.cse.unt.edu). He obtained his Ph.D. from the University of South Florida, ranked among the top public Universities in the USA, in 2003. He obtained his masters in System Science and Automation from the highest ranked institute from India, the Indian Institute of Science, Bangalore, in 1999. He obtained B. Tech. (Honors) degree in Electrical Engineering from Orissa University of Agriculture and Engineering in 1995. His research is in "Design and CAD for Low-Power High-Performance Nanoscale VLSI". His research is funded by National Science Foundation and Semiconductor Research Corporation. Prof. Mohanty is an author of 120+ peer-reviewed top-notch journal and conference publications. The publications are well-received by the world-wide peers with a total of 900+ citations resulting in an H-index of 17 (from Harzing's Publish and Perish). He is an author of a book titled "*Low-Power High-Level Synthesis for Nanoscale CMOS Circuits*",

published in 2008. He serves on the program committee of several international conferences and editorial board of several international journals. He is the Steering Committee Chair of the International Symposium on Electronic System Design. He is a senior member of IEEE and ACM.

Dhiraj K. Pradhan currently holds a Chair in the Department of Computer Science, University of Bristol, Bristol, U.K. He was a Professor with the Department of Electrical and Computer Engineering, Oregon State University, Corvallis. Previous to this, he had held the COE Endowed Chair Professorship with the Department of Computer Science, Texas A&M University, College Station, also serving as founder of the Laboratory of Computer Systems. Prior to this, he held a Professorship with the University of Massachusetts, Amherst, where he also served as Coordinator of Computer Engineering. He has also served as coauthor and editor of various books, including Fault-Tolerant Computing: Theory and Techniques, Volumes I and II (Prentice-Hall, 1986), Fault-Tolerant Computer Systems Design (Prentice-Hall, 1996, second print 2003), and IC Manufacturability: The Art of Process and Design Integration (IEEE Press, 2000). Prof. Pradhan is a Fellow of ACM and Japan Society of Promotion of Science. He is a recipient of a Humboldt Prize, Germany and the Fulbright-Flad Chair in computer science in 1997.

Maciej J. Ciesielski received the M.S. degree in electrical engineering from Warsaw Polytechnic, Warsaw, Poland, in 1974 and the Ph.D. degree in electrical engineering from the University of Rochester, Rochester, NY, in 1983. From 1983 to 1986, he was a Senior Research Engineer with GTE Laboratories on the silicon compilation project. In 1987, he joined the University of Massachusetts, Amherst, where he is a Professor and the Associate Department Head of the Dept. of Electrical and Computer Engineering. He teaches and conducts research in the area of electronic design automation, and specifically, in logic and high-level synthesis, optimization, and verification of digital systems. Dr. Ciesielski is the recipient of the Doctorate Honoris Causa from the Université de Bretagne Sud, Lorient, France, for his contributions to the field of Electronic Design Automation.