A Taylor Expansion Diagram Approach for Nano-CMOS RTL Leakage Optimization

S. Banerjee*, J. Mathew*, D. K. Pradhan*, S. P. Mohanty†, and M. Ciesielski‡
* University of Bristol, UK.
E-mail: shibaji@cs.bris.ac.uk
† University of North Texas, Denton, TX 76203, USA.
E-mail: saraju.mohanty@unt.edu
‡ University of Massachusetts, 309B Knowles Engineering Building, Amherst, USA.
E-mail: ciesiel@ecs.umass.edu

***Abstract*—Due to exponential behavior of gate-oxide leakage current with temperature and technology scaling, leakage power plays important role in $nano - CMOS$ circuit. In this paper, we present simultaneous scheduling and binding algorithm for optimizing leakage current during behavioral synthesis. It uses TED (Taylor Expansion Diagram) for generating optimized DFG (Data Flow Graph). Once DFG is obtained, it selectively binds non-critical components to corresponding functional unit consisting of transistors of high oxide thickness and critical components with low oxide thickness. As the algorithm considers time-constraint explicitly, it reduces leakage current without degrading the performance of the design. Experimental results on a set of behavioral synthesis benchmarks for $45nm$ process show 30% to 70% reduction in leakage current compared to the results obtained by a conventional optimization flow.**

I. Introduction

As $CMOS$ technology continues to scale down to achieve higher performance and higher level of integration, power dissipation poses new and difficult challenges for integrated circuit designers. While the initial works to reduce the power dissipation was to decrease the supply voltage, it quickly became apparent that this approach was insufficient. Designer subsequently began to focus on different methodology to tackle the power issues. The power dissipation in $CMOS$ circuit can be expressed as a sum of switching and leakage power as follows,

$$P = P_{switching} + P_{leakage} = \alpha.f.C.V_{dd}^2 + I_{leakage}.V_{dd} \quad (1)$$

Where, V_{dd} is supply voltage, α is switching activity, f is the clock frequency, C is the average switched capacitance of the circuit, and $I_{leakage}$ is the average leakage current. Most of the existing works have only considered dynamic power ($P_{switching}$) reduction for low-power behavioral synthesis. Some works on multiple threshold and power supply voltage assignment (multi V_{dd}/V_{th}) have been shown as an effective way to reduce the circuit power dissipation [1], [2], [3], [4]. However, these techniques do not reduce the leakage on the critical path and degrade the design yield under process variation [5]. On the other hand, the leakage power is responsible for significant portion of power dissipation because it is not only important in standby mode but also in the active mode of operation. Thus, a low power behavioral synthesis methodology must target reduction of the leakage power.

The gate-oxide leakage current I_{ox} in $CMOS$ is proportional to the square of supply voltage and inversely proportional to the square of T_{ox} (gate-oxide thickness). Reducing supply voltage will increase the delay of the circuit and hence would affect the performance of the design. The leakage current reduction based on $dual - V_{dd}$ can be found at [6]. However, $dual - V_{dd}$ requires extra power supply voltages and is not applicable in performance-critical circuit. It also increases the number of critical paths in a design which reduces the design yield under process variation. 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 which is less susceptible under process variation. In [7], authors have used $dual - T_{ox}$ based $CMOS$ technology to minimize the leakage current during behavioral synthesis. However, their RTL generation is not optimal. In our present work, we have used TED and STA based optimized techniques to generate optimal RTL at the end of the synthesis process.

In the paper, we address reduction of total gate-oxide leakage of a $CMOS$ data path circuit during HLS (high-level synthesis). In this work, we have used $TEDs$ (Taylor Expansion Diagrams) representation for high-level design description [8], [9], [10]. This representation is useful for modeling and supporting equivalence verification of designs specified at the behavioral level. TED is a canonical, graph based representation, similar to $BDDs$ (binary decision diagrams) [11] and $BMDs$ (binary moment diagrams) [12]. In contrast to $BDDs$ and $BMDs$, TED is based on a non-binary decomposition principle, modeled along the Taylors series expansion. TED is capable of capturing an entire class of structural solutions, rather than a single DFG (data flow graph). By using decomposition, TED can be converted into a structural representation, DFG, optimized for a particular design objective. After obtaining DFG, each of its nodes is scheduled at appropriate control step, and simultaneously bound them to the best available resources to achieve the desire performance with minimum gate-oxide leakage.

II. NANO $CMOS\ RTL$ OPTIMIZATION: THE PROBLEM AND THE PROPOSED SOLUTION

Power reduction in general can be achieved at various levels of design abstraction, such as system architecture (e.g., behavioral, high-level, algorithm), logic and transistor level. At each level of design abstraction researchers have proposed different techniques for reduction of various sources of power dissipation. Works on low-power HLS can be found at [1], [13], [14]. These techniques have been successfully implemented, but most of these works focused on one side of the issues of isolation. In [15], [16], $dual - T_{ox}$ is used for tunneling current reduction at logic or transistor level. Nevertheless, low power exploration for behavioral synthesis is still in its infancy. In this work, we describe $nano - CMOS\ RTL$ optimization technique for effectively reducing leakage current. This section formulates the objectives as an optimization problem, and then highlights contributions of this paper.

A. Problem definition:

The first task is to generate an optimized DFG from a given polynomial or circuit description. For this purpose, we focus on behavioral optimization based on $TED - based$ transformation and its functional decomposition, resulting in a construction of DFG. The DFG thus obtained is optimized in terms of the on number of components.

Once a DFG is obtained, next task is to perform STA (static timing analysis) to find critical paths in the design. Once critical components are identified, we can use appropriate scheduling and resource binding algorithms to minimize the total gate-oxide leakage current without degrading the circuit performance. This problem can be stated as follows,

Given an unscheduled DFG $G(V, E)$, perform STA to determine the critical and non-critical components. After that it is required to schedule the graph with appropriate binding algorithm such that the total gate-oxide leakage current is minimized and resource constraint (silicon cost) and delay constraint (circuit performance) are satisfied.

B. Contribution of the paper:

The contribution of the paper can be summarized as follows,

1) Given a circuit described as polynomial, generate a DFG by using appropriate TED optimization techniques.
2) Perform low-leakage behavioral synthesis which reduces the gate-oxide leakage dissipation of the circuit.
3) Apply $STA - based$ scheduling and resource binding algorithm with the objective to minimize gate leakage of datapath circuits using resources of different oxide thickness.

III. THE PROPOSED METHODOLOGY FOR $nano - CMOS\ RTL$ OPTIMIZATION

The behavioral synthesis flow for gate-oxide leakage minimization is shown in *Fig. 1*. The basic idea behind the proposed system is to transform the functional TED representation of the design to a structural DFG representation. DFG is obtained from TED by performing successive decomposition of TED by means of cuts [8]. The cut-based decomposition is guided in such a way as to optimize the DFG for a given objective. After obtaining DFG, STA is performed to identify the critical and non-critical components. By using simultaneous scheduling and binding approach on a partially scheduled DFG we can achieve more flexibility while binding resources to operations. The behavioral scheduling-binding algorithm using leakage and propagation delay estimator generates a circuit which dissipates minimal gate-oxide leakage. The delay-current estimator uses the pre-characterized $multi - T_{ox}$ datapath library and calculates the total gate-oxide leakage current and critical path delay of the circuits for a given DFG. Finally, RTL description of leakage-performance optimal datapath and control circuits are generated. The following subsection briefly describes about our $TED - based$ optimization approach.

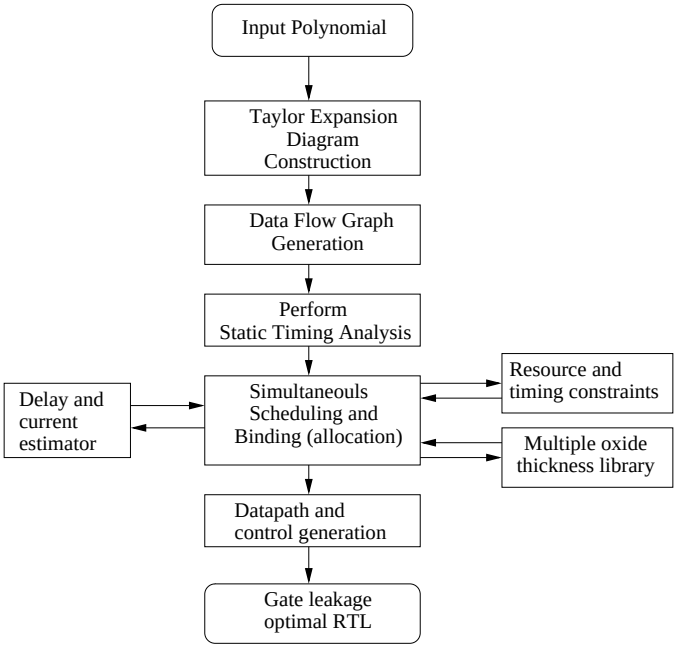

Fig. 1. The behavioral synthesis flow for gate-oxide lekage reduction

A. Canonical TED for Efficient High-Level Representation

Taylor Expansion diagram [9] 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 $f(x, y, ..)$, can be represented using Taylor series expansion, w.r.t. variable x as follows:

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

Where $f'(x)$, $f''(x)$, etc, are the successive derivatives of f w.r.t. 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. *Fig. 2a* shows one-level decomposition of

function $f(x, y, ..)$ at variable x. The nodes $f(x = 0, y, ..)$, $f''(x = 0, y, ..)$, etc, represent subsequent derivative functions that depends on the remaining variables. *Fig. 2b* shows TED for the function $f(A, B, C) = A^2 + AB + 2AC + 2BC$. The detailed explanation of TED can be found in [8], [9], [10].

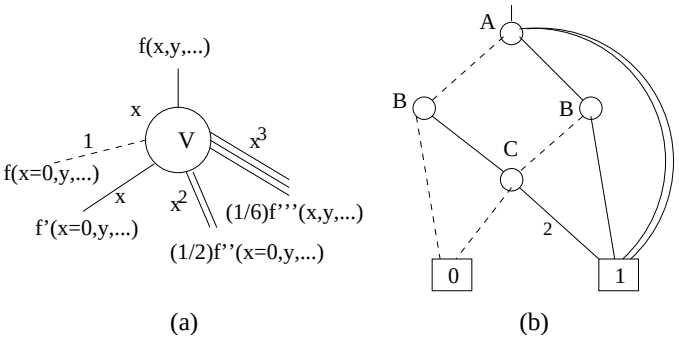

Fig. 2. TED [17]: a. Decomposition principle; b. TED example for $f(A, B, C) = A^2 + AB + 2AC + 2BC$

B. $TED - based\ RTL$ low-leakage optimization: A Finite Impulse Filter (FIR) Case Study

Since FIR (Finite-impulse response) filters are critical to most DSP application, an energy-aware filter design helps significantly in reducing the total power dissipation. The polynomial corresponding to a $4 - tap\ FIR$ filter can be written as,

$$Y[n] = a_0X[n] + a_1X[n-1] + a_2X[n-2] + a_3X[n-3] \quad (3)$$

or equivalently as,

$$Y_n = a_0X_n + a_1X_{n-1} + a_2X_{n-2} + a_3X_{n-3} \quad (4)$$

where $X_n = X[n]$, $X_{n-1} = X[n-1]$, $X_{n-2} = X[n-2]$, and $X_{n-3} = X[n-3]$.

TED corresponding to equation 5 is shown in *Fig. 3* and the optimized TED is shown in *Fig. 4*. Given an optimized TED, the next task is to convert it to DFG, shown in *Fig. 5*. An STA on DFG is performed to generate the necessary timing information. Specifically, we need to calculate arrival time T_a, required time T_r, and slack $T_s = T_r - T_a$, for each node.

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:

$$T_a(n) = Delay(n) + max(T_a(n_i)|_{n_i \in Input(n)}) \quad (5)$$

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:

$$T_r(n) = min(T_r(n_o)|_{n_o \in output(n)}) - Delay(n) \quad (6)$$

Here $Output(n)$ is the set of output DFG nodes of node n

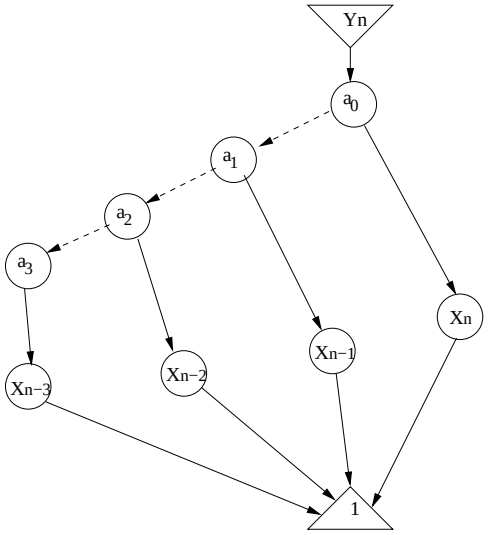

Fig. 3. TED for a $4 - tap\ FIR$ filter

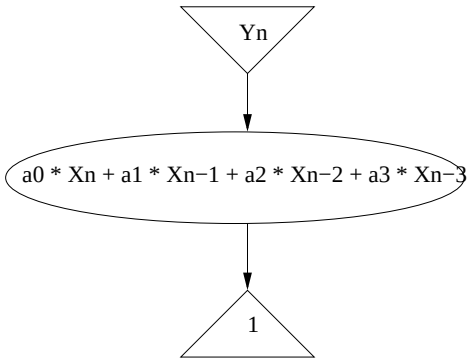

Fig. 4. Optimized TED for equation 5

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.

$$T_s(n) = T_r(n) - T_a(n) \quad (7)$$

In *Fig. 5*, the arrival time T_a, the required time T_r, and the slack T_s of each node are denoted in the form of $[T_a/T_r/T_s]$. Here, we assume delay of each functional unit is 1 for simplicity. Based on the definition of slack, a critical node

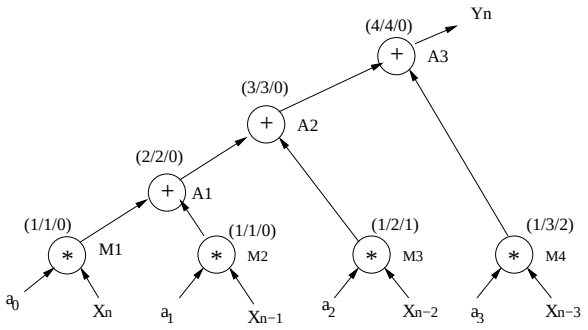

Fig. 5. DFG for the TED of *Fig. 4*

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. 5*, critical path 1 consists of 4 nodes ($M1$, $A1$, $A2$, $A3$) and critical path 2 consists of 4 nodes ($M2$, $A1$, $A2$, $A3$). However, nodes $M3$ and $M4$ have non-zero slack. So, they can be bound to the library having high gate-oxide thickness to reduce the gate-oxide leakage, provided it should not violate the slack requirement. In other words, the slack of these nodes should not be negative after binding to the higher gate-oxide thickness library. All the nodes in the critical path will map to the low gate-oxide thickness library to reduce the latency of the design as much as possible. Thus, even if these nodes or FUs (functional unites) are affected by process variation, performance of the design would not be affected much. In the next subsection, we present the generalized algorithm for simultaneous scheduling-binding for general circuits.

C. An Algorithm for $Nano-CMOS$ RTL leakage optimization

In this section, we present a leakage optimization algorithm for simultaneous scheduling and binding under resource constraint. The inputs to the algorithm are an unscheduled DFG, libraries with different recourses made of transistors of different oxide thickness, and a delay trade-off factor T_d. The T_d is a user defined quantity which specifies the maximum allowed critical path delay of the targeted circuit. The algorithm schedules and binds the nodes of DFG to the FUs of different libraries so that critical path delay is either equal or less than T_d while at the same time gate-oxide leakage current of the target circuit should be minimized.

The proposed time-resource constrained algorithm (Algorithm_1) takes time constraint T_d as an input. It performs a STA on the DFG and identifies critical and non-critical nodes by calculating T_a, T_r, and T_s of each nodes. During the step, it uses delay value of 1 for each node. Once identified, it assigns T_{ox_L} (FUs from low thickness gate-oxide library) to critical nodes and T_{ox_H} to non-critical nodes. After initial scheduling and binding, it calculates the critical path delay. If critical path delay is less than T_d, the algorithm checks individual nodes which were assigned to T_{ox_L}. It replaces the T_{ox_L} with T_{ox_H} to reduce the leakage current. If T_{ox_H} is not available at that control step, it schedules it to next available control step under the condition that replacement should not violate the timing property.

Consider the FIR filter of *Fig. 5* under the assumption that unlimited number of T_{ox_L} and T_{ox_H} components and $T_d = 6$ ns. We also assume that delay of the adder and multiplier corresponding to T_{ox_H} library are 2 ns and 3 ns respectively, while those corresponding to T_{ox_L} library are 1 ns and 2 ns. After identifying the critical and non-critical nodes, the present algorithm replaces the critical components with T_{ox_L} and non-critical to T_{ox_H} respectively. Nodes $A1$, $A2$, $A3$, $M1$ and $M2$ are assigned to the corresponding components of T_{ox_L} and nodes $M3$ and $M4$ are bound to T_{ox_H}. After initial scheduling

Algorithm 1 leakage optimization for $Nano-CMOS$

```
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 n_i do
  if FU_j(k, T_ox_L) is available for control step C[n_i] then
    Assign FU_j(k, T_ox_L) to node n_i
  else
    Assign FU_j(k, T_ox_H) 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 FU_j(k, T_ox_H) is available for control step C[n_i] then
      schedule n_i in control step C[n_i]
      Assign FU_j(k, T_ox_H) 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 FU_j(k, T_ox_L) is available for control step C[n_i] then
        schedule n_i in control step C[n_i]
        Assign FU_j(k, T_ox_L) 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
Sort all critical nodes according to ascending order of leakage current
for all critical nodes n_i do
  if FU_j(k, T_ox_H) is available for control step C[n_i] then
    Assign FU_j(k, T_ox_H) to node n_i
    if slack of n_i is less than 0 then
      Assign FU_j(k, T_ox_L) to node n_i
    else
      update T_a, T_r, T_s for all nodes connected to n_i
      calculate critical path delay T_cp
      if T_cp greater than T_d then
        Assign FU_j(k, T_ox_L) to node n_i
      end if
    end if
  end if
end for
```

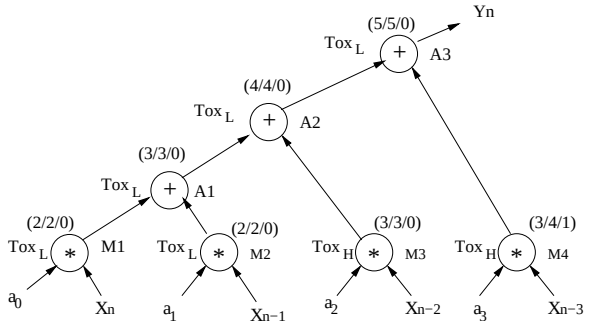

Fig. 6. DFG for the TED of *Fig. 4* after initial scheduling and binding

and binding, the algorithm calculates T_a, T_r, and T_s for all the nodes. The value of T_a, T_r, and T_s after initial scheduling and binding is shown in *Fig. 6*. In *Fig. 6*, the delay of the critical path is $5\ ns$, which is less than T_d ($6\ ns$). So, the algorithm checks to replace the node T_{ox_L} for further reduction of leakage current if and only if replacement does not cause any timing violation. It is easy to see from *Fig. 6* that the $A3$ can be replaced by T_{ox_H} without causing any timing violation; the corresponding DFG is shown in *Fig. 7*.

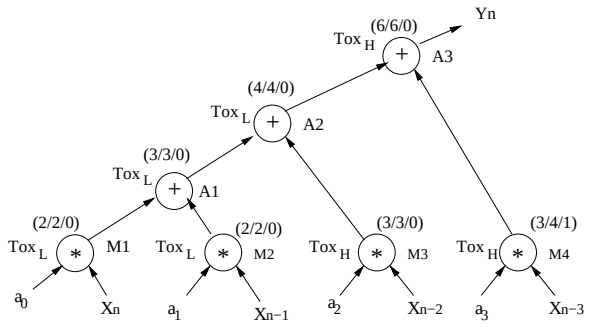

Fig. 7. Final DFG for *Fig. 4*

IV. Experimental Results

The above algorithm, TED, and STA are implemented in C. Our system does not need any other external tool for synthesis. Experiments were performed on several behavioral level benchmark circuits with several constraints. The resource constraints are expressed as the functional units of different oxide thickness and time constraints in term of delay trade-off factor (T_d). The goal of the experiments is to demonstrate (i) the reduction of leakage current without violating system performance, (ii) Output synthesized netlist of a given design is less susceptible under process variations.

In order to perform experiment, we first need to set up the library with different gate-oxide thickness. In the present work, we characterized a library of $16-bit$ datapath components, such as adder, subtractors, multipliers, divider, multiplexers, and registers following the structural descriptions from [18]. *Fig. 8* shows variation of I_{ox} leakage and propagation delay for the multiplier with respect to T_{ox}. It is clear from the figure that I_{ox} is almost 23 times lower when T_{ox} increases

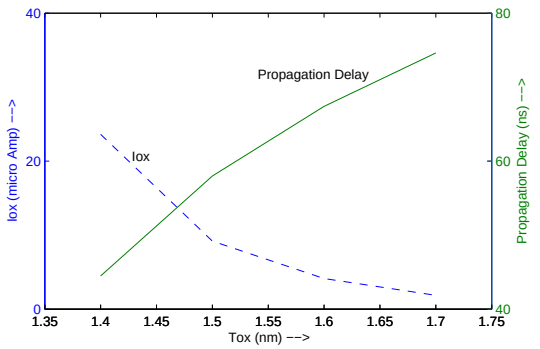

Fig. 8. Variation of I_{ox} and delay w.r.t. T_{ox}

TABLE I
Library with different gate-oxide thickness

Functional unit	$T_{ox} = 1.4nm$		$T_{ox} = 1.7nm$	
	$I_{ox}\ (\mu A)$	$T_{pd}\ (ns)$	$I_{ox}\ (\mu A)$	$T_{pd}\ (ns)$
Adder	1.765620	27.916601	0.13848	46.82190
Subtractor	1.973340	27.916601	0.15579	46.82190
Multiplier	23.622379	44.484201	1.86948	74.62210
Divider	36.397161	151.16479	2.88500	253.55799
Comparator	4.189020	35.860901	0.32889	60.14969
Register	1.402110	32.679299	0.10963	54.82440
Multiplexer	1.194390	1.581100	0.09232	2.65780

from $1.4nm$ to $1.7nm$ and corresponding propagation delay is almost doubled for the same change. Due to this reason we first setup a library of dual-oxide thickness pair of $1.4nm-1.7nm$, shown in *Table I*. *Table I*, I_{ox} and T_{pd} represent the leakage current and propagation delay of the functional unit, respectively, for a given gate-oxide thickness. For each benchmark, we present gate-leakage current for different T_d. We also used a smaller number of T_{ox_H} resources and high number of T_{ox_L} resources. The results are shown in *Table II*. The factor I_{ox_S} represents the gate-oxide leakage current when only T_{ox_L} library ($1.4nm$ oxide thickness) is used for the total design. The percentage reduction in gate-oxide leakage current is calculated as,

$$\triangle I = \frac{I_{ox_S} - I_{ox}}{I_{ox_S}} * 100 \tag{8}$$

Table II shows the results of the our scheduling algorithm. Column 2 in *Table II* represents the number of available T_{ox_H} resources in the library. The results indicate reduction in gate leakage current in the range of 30% to 70% when number of T_{ox_H} resources increases from 1 to unlimited number. *Fig. 9* shows the average percentage reduction for all benchmarks without resource constraints. Results indicate high leakage current reduction without degrading system performance.

V. Conclusions

In this paper, we presented scheduling-binding algorithm for reducing gate-oxide leakage current using $dual-T_{ox}$ approach. The algorithm is based on TED for generating optimized DFG on which proposed algorithm is applied. Experimental results on a set of benchmark circuits show promising results in terms of leakage power saving.

TABLE II
EXPERIMENTAL RESULTS FOR THE PRESENT ALGORITHM

Circuits	resource cons	I_{ox_L} (μA)	$T_d = 1.0$ (ns)		$T_d = 1.2$ (ns)		$T_d = 1.4$ (ns)		$T_d = 1.6$ (ns)	
			I_{ox} (μA)	ΔI	I_{ox} (μA)	ΔI	I_{ox} (μA)	ΔI	I_{ox} (μA)	ΔI
ARF	1	399.146	326.501	18.2	319.31	20.3	306.54	23.2	298.56	25.2
	2	399.146	275.74	31.1	269.42	32.5	262.63	34.2	255.05	36.1
	3	399.146	259.44	34.8	251.06	37.1	241.88	39.4	228.71	42.7
	∞	399.146	145.30	64.8	139.70	65.9	126.53	68.3	107.77	73.1
	Average I_{ox} Reduction			37.2	$--$	38.9	$--$	41.3	$--$	44.3
BPF	1	271.64	196.67	27.6	193.14	28.9	189.06	30.4	180.09	33.7
	2	271.64	124.41	54.2	120.34	55.7	115.99	57.9	108.66	60.2
	3	271.64	115.99	57.2	114.36	57.9	112.73	58.5	100.51	63.7
	∞	271.64	104.03	61.7	103.92	61.9	98.61	63.7	96.23	65.1
	Average I_{ox} Reduction			50.2	$--$	51.1	$--$	52.6	$--$	55.7
FIR	1	215.463	187.33	13.9	181.42	15.8	179.48	16.7	176.25	18.2
	2	215.463	160.95	25.3	159.23	26.1	155.13	28.2	149.53	30.6
	3	215.463	100.83	53.2	96.31	55.3	90.49	58.1	86.61	59.8
	∞	215.463	86.18	60.7	81.87	62.1	75.41	65.2	74.55	66.4
	Average I_{ox} Reduction			38.3	$--$	39.8	$--$	42.1	$--$	43.8
EWF	1	234.885	210.51	10.4	202.47	13.8	201.53	14.2	196.36	16.4
	2	234.885	191.43	18.6	187.91	20.4	180.86	23.5	176.16	25.1
	3	234.885	162.07	31.2	160.89	31.5	152.68	35.1	150.33	36.7
	∞	234.885	143.28	39.5	136.23	42.3	133.88	43.4	129.19	45.1
	Average I_{ox} Reduction			24.9	$--$	27.1	$--$	20.1	$--$	30.8
DWT	1	204.87	190.24	7.1	186.43	9.7	184.38	10.4	180.28	12.3
	2	204.87	175.14	14.5	171.68	16.2	170.05	17.9	163.89	20.1
	3	204.87	156.21	23.7	149.35	27.1	145.46	29.2	140.13	31.6
	∞	204.87	135.57	33.8	132.76	35.2	127.02	38.1	122.92	40.2
	Average I_{ox} Reduction			19.8	$--$	22.1	$--$	23.9	$--$	26.1

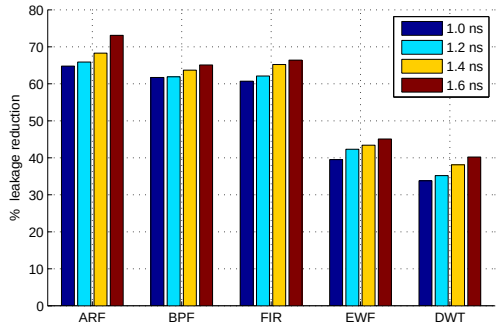

Fig. 9. Bar chat shows percentage reduction of leakage current for different T_d under no resource constraints

REFERENCES

[1] 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.

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

[3] A. Srivastava, and D. Sylvester "Minimizing total power by simultaneous Vdd/Vth assignment," IEEE TCAD, 2004, vol. 23, pp 665 – 677.

[4] K. Usami, and M. Igarashi "Low-power design methodology and applications utilizing dual supply voltages," proc. ASPDAC, 2000, pp 123 – 128.

[5] M. Liu, Wang Wei-Shen, and M. Orshansky "Leakage Power Reduction by Dual-Vth Designs Under Probabilistic Analysis of Vth Variation," Proc. International Symposium on Low Power Electronics and Design, 2004, pp 2 – 7.

[6] R.K. Krishnarnurthy, A. Alvandpour, V. De and S. Borkar "High-performance and low-power challenges for sub-70 nm microprocessor circuits," Proc. IEEE Custom Integrated Circuits Conference, 2002, pp 125 – 128.

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

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

[9] 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.

[10] M. Ciesielski, P. Kalla, Zhihong Zheng, and B. Rouzeyre "Taylor expansion diagrams: a compact, canonical representation with applications to symbolic verification," Proc. DATE, 2002, pp. 285 – 289

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

[12] R.E. Bryant, and Y.-A Chen "Verification of Arithmetic Circuits with Binary Moment Diagrams," proc. 32nd Conference on Design Automation, 1995, pp. 535 – 541.

[13] V. Krishnan and S. Katkoori "Simultaneous Peak Temperature and Average Power Minimization during Behavioral Synthesis," Proc. 22nd International Conference on VLSI Design, 2009, pp. 419 – 424.

[14] Insup Shin, Seungwhun Paik, and Youngsoo Shin "Register allocation for high-level synthesis using dual supply voltages," Proc. 46th ACM/IEEE DAC, 2009, pp. 937 – 942.

[15] N. Sirisantana, and K. Roy "Low-power design using multiple channel lengths and oxide thicknesses," IEEE Design and Test of Computers, Jan-Feb 2004, vol. 21, pp. 56 – 63.

[16] A.K. Sultania, D. Sylvester, and S.S. Sapatnekar "Gate oxide leakage and delay tradeoffs for dual-Tox circuits," IEEE Transactions on VLSI Systems, Dec. 2005, vol. 13, pp. 1362 – 1375.

[17] M. Ciesielski, S. Askar, D. Gomez-Prado, J. Guillot, and E. Boutillon "Data-Flow Transformations using Taylor Expansion Diagrams," Proc. DATE, 2007, pp. 1 – 6.

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