Simultaneous Scheduling and Binding for Low Gate Leakage Nano-CMOS Datapath Circuit Behavioral Synthesis

Saraju P. Mohanty
Computer Science and Engineering
University of North Texas, Denton, TX 76203.
Email: smohanty@cse.unt.edu
Elias Kougianos
Electrical Engineering Technology
University of North Texas, Denton, TX 76203.
Email: eliask@unt.edu
Dhiraj K. Pradhan
Department of Computer Science
University of Bristol, Bristol, UK.
Email: pradhan@cs.bris.ac.uk

Abstract

In this paper we present two polynomial time-complexity heuristic algorithms for optimization of gate-oxide leakage (tunneling current) during behavioral synthesis through simultaneous scheduling and binding. One algorithm considers the time-constraint explicitly and the other considers it implicitly while both account for resource constraints. The algorithms selectively bind the off-critical operations to instances of the pre-characterized resources consisting of transistors of higher oxide thickness, and critical operations to the resources of lower oxide thickness for power and performance optimization. We design and characterize functional and storage units of different gate-oxide thicknesses and built a datapath library. Extensive experiments for several behavioral synthesis benchmarks for $45nm$ CMOS technology showed that reduction as high as 85% can be obtained.

Index Terms

Nano-CMOS, Gate-Oxide Leakage, Direct Tunneling, Multiple-T_{ox} Technology, Low Power Synthesis

I. Introduction

Several issues such as battery life, reliability, thermal considerations, and environmental concerns have driven the need for low power designs [1]. Aggressive scaling of CMOS devices has been taking place to address the increasing market demand for smaller and feature packed portable electronic devices. In such scaled technologies, both static and dynamic power have become equally contributing factors for the total power dissipation of a CMOS circuit [2], [3]. Each of the contributing leakage currents has several forms and origins and they flow between different terminals and during different operating conditions of a nano-CMOS transistor.

This work was partially supported by NSF award number 0702361.
This journal paper is based on our following shorter peer reviewed conference versions:
S. P. Mohanty and E. Kougianos, "Modeling and Reduction of Gate Leakage during Behavioral Synthesis of NanoCMOS Circuits", in *Proceedings of the 19th IEEE International Conference on VLSI Design (VLSID)*, pp. 83-88, 2006.
S. P. Mohanty, et. al., "Scheduling and Binding for Low Gate Leakage NanoCMOS Datapath Circuit Synthesis", in *Proceedings of the 38th IEEE International Symposium on Circuits and Systems (ISCAS)*, pp. 5291-5294, 2006.

In a short channel nanometer CMOS transistor, several forms of leakage current exist, such as reverse biased diode leakage, subthreshold leakage, gate-oxide tunneling current, hot carrier gate current, gate induced drain leakage and channel punch through current [3]. Of all these leakage mechanisms, gate-oxide (direct) tunneling current that flows during both active and sleep modes of a device is the most significant component for low-end nano-CMOS technology of $65nm$ and below. Thus, the major sources of power dissipation in a nano-CMOS circuit can be summarized as dynamic power and subthreshold and gate-oxide leakage.

In this work, we address reduction of total gate-oxide leakage of a CMOS datapath circuit during behavioral synthesis. We incorporate time constraints as a performance (or delay) trade-off factor and offer the user with the choice of predetermining the performance of a circuit vis-a-vis the power requirements. The algorithms consider an unscheduled data flow graph (DFG), schedule each of its nodes at appropriate control steps, and simultaneously bind them to the best available resource while considering resource constraints so as to achieve the desired performance with the minimum gate-oxide leakage.

The rest of the paper is organized as follows. Related research works are summarized in section II. In section III, we present motivations, define the gate leakage reduction as an optimization problem, and summarize our contributions. The proposed simultaneous scheduling-binding algorithms are elaborated in section IV. We describe a design methodology for a standard cell library in terms of total gate leakage and propagation delay in section V. Experimental results are presented and discussed in section VI. Our findings, conclusions and suggestions for future research are summarized in section VII.

II. Related Research

Power reduction in general can be achieved at various levels of design abstraction, such as system, architectural (*aka* behavioral, algorithmic, high-level), logic and transistor level. At each level of design abstraction researchers have proposed different techniques for reduction of various sources of power dissipation. For brevity, we discuss selected existing research works that use scaling of threshold voltage (V_{th}) or oxide thickness (T_{ox}) as the approach for leakage and performance tradeoffs. In [4], [5], [6], algorithms have been proposed for subthreshold leakage reduction using dual-V_{th} library during behavioral synthesis. In [7], [8], dual-T_{ox} is used for tunneling current reduction at logic or transistor level.

The existing literature has many mature research works addressing dynamic and subthreshold power dissipation, but research addressing gate-oxide leakage is still in full swing. In this paper we focus on the reduction of gate-oxide leakage of nano-CMOS datapath circuits during *behavioral* synthesis, as opposed to existing approaches which are at transistor level.

This archival journal version is based on our previous conference publications [9], [10]. In this paper emphasis is given on two proposed algorithms and their comparative results. All the results are presented for $45nm$ CMOS technology. The characterization methodology is elaborated. Trends of gate-oxide leakage and delay with respect to gate-oxide thickness are presented. Gate-oxide leakage reduction is presented for various time and/or resource constraints for design space exploration.

III. Motivation, Problem Definition, and Contributions

In this section we present the motivation for using the dual-T_{ox} based CMOS technology, formulate the objectives as an optimization problem, and then highlight contributions of this paper.

A. Motivation

It is evident from the related research available in the existing literature that low-power behavioral synthesis research works have mostly considered dynamic power reduction. Only few behavioral synthesis works address subthreshold leakage. There is no behavioral synthesis work available in the current literature to optimize gate leakage. This calls for a new behavioral synthesis approach considering gate leakage for the low-end nanoscale technology ($65nm$ and below) domain. Thus, we propose simultaneous scheduling and binding using dual-T_{ox} resources to minimize the gate-oxide leakage of the overall datapath circuit while keeping the performance degradation under control. While performance can be incorporated as time constraints in the form of a delay trade-off factor (or implicitly, through a current-delay-product), the cost of silicon is taken into account as a resource constraint.

Why Gate-Oxide Leakage Reduction During Behavioral Synthesis?: The behavioral level is not as highly abstracted as the system level nor as detailed as the logic-gate or transistor level. Hence, at behavioral level there is a balanced degree of freedom to explore power reduction mechanisms and it can help in investigating lower power design alternatives prior to circuit layout in actual silicon. Moreover, correct design decisions at early phases of

circuit abstraction (like behavioral level) will ensure that design errors are not propagated to lower levels, which may be costly to rectify.

Why Multi-T_{ox} Approach during Behavioral Synthesis?: The probability of gate-oxide tunneling is a strong function of the barrier height and the barrier thickness. For supply voltage V_{dd}, and effective gate oxide thickness T_{ox}, the gate-oxide leakage dissipation in a CMOS can be described as: [11]:

$$I_{ox} \propto \left(\frac{V_{dd}}{T_{ox}}\right)^2 exp\left(-\gamma \frac{T_{ox}}{V_{dd}}\right), \quad (1)$$

where γ is an experimentally derived factor. Based on Eqn. 1, we have the following possible options for reduction of gate leakage: (i) decreasing the supply voltage and (ii) increasing the gate oxide thickness. Decreasing the power supply voltage is used as a popular option to reduce dynamic power consumption [12], [13] which will play its role in the reduction of leakage power as well, but it may not be sufficient to control the exponential growth of gate leakage. Increase in the gate-oxide (SiO_2) thickness leads to increase in propagation delay. Thus, we conclude that use of multiple gate-oxide thicknesses can serve as a leakage power (current) and performance (delay) trade-off.

B. Problem Definition

Given an unscheduled data flow graph (UDFG) $G(V, E)$, it is required to find the scheduled data flow graph (SDFG) with appropriate resource binding such that the total gate-oxide leakage current is minimized and resource constraints (silicon cost) and latency constraints (circuit performance) are satisfied.

The above can be stated as an optimization problem as follows. Let V be the set of all vertices and V_{CP} be the set of vertices in the critical path from the source S of the DFG to the output or sink node D. For simplicity, we assume that the DFG has a single source node and a single sink node. However, the formulation can be extended to multiple source, multiple sink graphs, whereby we will have corresponding response times for each source and sink pair. The cost (resource constrained) and performance (latency constrained) driven tunneling current minimization problem can thus be formulated as follows:

$$\text{Minimize} \quad \sum_{v_i \in V} I_{ox}(v_i), \quad (2)$$

where, $I_{ox}(v_i)$ is the tunneling current consumed per sample node v_i of the DFG, such that the following resource

and latency constraints, respectively, are satisfied:

$$\sum_{v_i \in V_{CP}} D_i(v_i) \quad \leq \quad D_{CP} \quad \forall \; v_i \in V_{CP}, \tag{3}$$

$$\text{Allocated } \left(FU_i\left(k, T_{ox}\right)\right) \quad \leq \quad \text{Available } \left(FU_i\left(k, T_{ox}\right)\right). \tag{4}$$

The constraints in Eqn. (3) ensure that the summation of all delays $D_i(v_i)$ is less than the critical path delay D_{CP}. The resource allocation is summarized in Eqn. (4), where the total allocation of the i^{th} resources (functional units) of type k and made up of transistors of oxide thickness T_{ox} denoted as $\left(FU_i\left(k, T_{ox}\right)\right)$ should be less than the total number of corresponding resources available.

C. Contributions of this Paper

The contributions of this paper to the state-of-the-art can be summarized as follows:

- Behavioral level minimization of gate-oxide leakage dissipation of datapath circuits accounting for both ON and OFF states of the device. Existing gate-oxide leakage reduction techniques are in logic or circuit level and account primarily for the OFF state, but not both ON and OFF states.
- Introduction of two heuristic algorithms that simultaneously perform operation scheduling and resource binding with the objective of gate leakage minimization of datapath circuits using resources of different oxide thickness.
- Consideration of both resource and time constraints (implicitly or explicitly) to provide the user with the flexibility to explore the design space of gate-oxide leakage power, circuit performance, and cost of silicon.
- Development of a multiple oxide thickness datapath component library containing functional and storage units.
- Proof that the NAND logic gate has minimal gate-oxide leakage and delay compared to other fundamental logic gates for the same number of inputs.
- Study of trends of gate leakage and propagation delay of architectural units with respect to oxide thickness.

IV. The Proposed Algorithms for Simultaneous Scheduling and Binding

The behavioral synthesis flow for gate-oxide leakage minimization is shown in Fig. 1. Various steps involved in the behavioral synthesis are: compilation, transformation, operation scheduling, resource (or functional unit) allocation, resource binding, connection allocation and architecture generation. Scheduling and binding are the major phases of low-power behavioral synthesis. During the compilation and transformation phases, behavioral

VHDL is compiled to structural VHDL to obtain a DFG. Scheduling time stamps the variables and operations in the DFG so that the operations in the same group can be executed concurrently. While allocation fixes the number and types of resources to be used in the datapath circuit, the binding process involves attaching operations to functional units and variables to memory units. With a conventional scheduling and binding approach we lose flexibility while binding is done. With a simultaneous scheduling and binding approach we can achieve more flexibility while binding resources to operations. The behavioral scheduling-binding algorithm using gate-oxide 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. The connection allocation step determines the types or number of buses, and buffers and multiplexors for the communication between resources. Finally, RTL descriptions of leakage-performance optimal datapath and control circuits are generated.

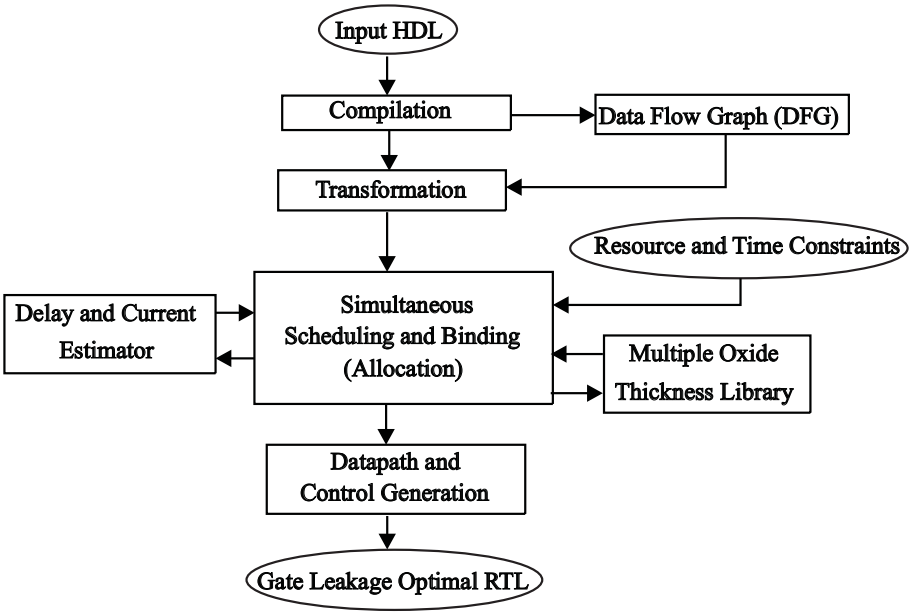


Fig. 1. The behavioral synthesis framework for gate-oxide leakage reduction.

We assume that the datapath is specified as a sequencing data flow graph (DFG), which is a directed acyclic DFG. Each vertex of the DFG represents an operation and each edge represents a dependency. Also, each vertex has attributes that specify the operation type. The delay of a control step is dependent on the delays of the functional unit, the multiplexer, and register. We assume that each node connected to the primary input is assigned two registers and one multiplexer while the inner nodes of the DFG have one register and one multiplexer.

We present two algorithms for simultaneous scheduling-binding. The inputs to the algorithms are an unscheduled DFG, a resource constraint matrix and a delay trade-off factor (T_F). The resource constraints include the number of different resources made of transistors of different oxide thickness. The delay trade-off factor (T_F) is a quantity that specifies the critical path delay of the target circuit with respect to the nominal case critical path delay. The first algorithm considers time constraints explicitly and the second algorithm considers time constraints implicitly through a current-delay product. The first algorithm can provide flexibility to the designer to provide time constraints as an input. The second algorithm has the advantage that it will converge to solutions faster as the time constraint is not stringent. The scheduling-binding algorithms generate various outputs, such as scheduled DFG with appropriate datapath functional unit assignment to an operation and estimates of gate-oxide leakage and delay. We assume that different functional units are characterized for gate-oxide leakage current and propagation delay for various gate-oxide thicknesses and are available in the datapath component library. All the transistors inside the same resource have the same gate-oxide thickness, and the transistor gate-oxide thickness may differ for various functional units. The algorithms can be easily used to handle various types of datapath operations, such as multicycling, chaining, and pipelining. For example, to take the multicycling operation into account the algorithm can assume the delay of the fastest unit as clock width and time stamp vertices needing slower unit to more than one clock steps.

A. Algorithm 1: Explicit Time Constrained and Resource Constrained

We present the proposed time-resource constrained heuristic algorithm that considers time constraints explicitly in Algorithm 1. The algorithm first performs a resource constrained ASAP schedule (C_S) and a resource constrained ALAP schedule (C_L) and identifies the set of nodes with zero and non-zero mobility. With the available resources, maximum gate-oxide leakage reduction can be achieved when resources with higher thickness (T_{ox_H}) are assigned to as many nodes as possible. Since the algorithm has no prior knowledge of the critical path, it initially assigns T_{ox_H} resources to zero mobility nodes. For each allowable clock cycle of the non-zero mobility nodes the algorithm identifies the clock cycle that can allocate a T_{ox_H} resource and schedules the node to that clock cycle and binds its operation with that T_{ox_H} resource. If such a clock cycle is not available, then it allocates a T_{ox_L} resource in the ASAP clock cycle and schedules it to that clock cycle. Once the scheduling and binding is done, the critical path delay is calculated. If the critical path delay violates the trade-off factor, then the algorithm tries to assign T_{ox_L}

resources by swapping the resources of the critical path node and the node containing T_{ox_L} resource in that clock cycle. If there is no T_{ox_L} resource active in that cycle then the algorithm assigns an available T_{ox_L} resource.

Algorithm 1 Explicit Time Constrained and Resource Constrained Simultaneous Scheduling and Binding Heuristic

Get resource constrained ASAP schedule C_S of the DFG.
Get resource constrained ALAP schedule C_L of the DFG.
Fix the number of control steps as maximum of the ASAP and ALAP schedules.
Classify the vertices in V to zero mobile and non-zero mobile vertices.
for all (Zero mobility vertex v_i in ASAP order) **do**
    Time stamp v_i in ASAP manner.
    if ($FU_j\,(k, T_{ox_H})$ is available for control step $C[v_i]$) **then**
        return Assign $FU_j\,(k, T_{ox_H})$ resource vertex v_i.
    else
        return Assign $FU_j\,(k, T_{ox_L})$ resource vertex v_i.
    end if
end for
for all (Non-zero vertex in ASAP order) **do**
    for all (Possible clock cycle $c : C_S[v_i] \rightarrow C_L[v_i]$) **do**
        if ($FU_j\,(k, T_{ox_H})$ is available for c) **then**
            return Schedule v_i in control step c and Assign $FU_j\,(k, T_{ox_H})$ resource vertex v_i.
        else if ($FU_j\,(k, T_{ox_L})$ is available for c and v_i is not already scheduled) **then**
            return Schedule v_i in control step c and Assign $FU_j\,(k, T_{ox_L})$ resource vertex v_i.
        end if
    end for
end for
Call Floyd's algorithm to determine critical path and corresponding vertices.
Calculate critical path delays for single-T_{ox} ($T_{CP_{ST}}$) and dual-T_{ox} ($T_{CP_{DT}}$).
for all (Control step c in the available schedule) **do**
    if (Time constraint not met i.e. $T_{CP_{DT}} > T_F \times T_{CP_{ST}}$) **then**
        return Find all vertices in a control step c.
        if (v_i is in the critical path and using T_{ox_H} resource) **then**
            return Check rest of the vertices in c.
            if (v_j is not in critical path and using T_{ox_L} resource) **then**
                return Swap the resources of v_i and v_j.
            else
                return Assign T_{ox_L} resource to v_i.
            end if
        end if
    end if
end for

A special case of the time-resource constrained algorithm is the case of time-constrained approach when we relax the constraints on the number of resources. In this case, the algorithm gives an RTL that has the least gate leakage while satisfying a time constraint. With this we can achieve maximum gate leakage reduction by assigning T_{ox_H} resources to all non critical path nodes. The algorithm in this case produces the minimal number of resources

required for execution of the scheduled DFG.

B. Algorithm 2: Implicit Time Constrained and Resource Constrained

We now present the algorithm that handles the time constraints implicitly, thus it can reduce the running time of the synthesis process. The combined reduction of gate-oxide leakage current dissipation and critical path delay translates to reduction of the gate-oxide leakage current-delay-product (CDP). Thus, the objective of the scheduling-binding algorithm is to minimize the CDP while assigning a schedule for the DFG. This implicitly facilitates minimization of the gate-oxide leakage current along with delay while considering resource constraints. If $I_{ox\,FU}(c, r)$ is the gate-oxide leakage current and $T_{pd_{FU}}(c, r)$ is the propagation delay of the r-th functional unit active in the control step c, then the gate-oxide leakage current-delay-product can be calculated as:

$$CDP \quad = \quad \sum_{c=1}^{N_c} \sum_{r=1}^{n_{FU\,c}} I_{ox\,FU}(c, r) * T_{pd_{FU}}(c, r), \tag{5}$$

The algorithm time stamps the operations such that low oxide thickness resources are active in the critical path and high-oxide thickness resources are active in the off-critical path of the datapath. The algorithm attempts to operate the higher intrinsic leakage units (such as multiplier and divider) of the higher oxide thickness to reduce the gate-oxide leakage and at the same time lower intrinsic leakage units (such as adder and subtractor) of lower oxide thickness to compensate for the delay increase as much as possible. This is in accordance with our observation from characterization that multiplier and divider units dissipate much more gate-oxide leakage compared to adder and subtractor units. At the same time it is observed that adder and subtractor units have lower delays compared to the multiplier and divider. The same assignment is carried out in the case of all potential off-critical paths and the CDP is calculated at each step and for each assignment for the DFG using Eqn. 5. Once the minimum CDP is obtained, a particular vertex is time stamped and the T_{ox} assignment is accepted.

The pseudocode of the scheduling-binding algorithm heuristic that considers time constraints implicitly is presented in Algorithm 2. The algorithm first determines the ASAP and ALAP schedules. While the ASAP schedule is unconstrained, the ALAP schedule uses the number of control steps found in the ASAP schedule as the latency constraint. The above obtained ASAP and ALAP schedules are modified using the resource constraints to determine the resource constrained ASAP and ALAP schedules. The resource constrained ASAP and ALAP schedules restrict the mobility of vertices to a great extent and reduce the solution search space for the heuristic. In the next step

the algorithm identifies the critical path V_c and the off-critical paths V_{oc}. For each critical vertex V_c the algorithm assigns the highest gate oxide thickness to the operation needing higher leakage units, while we assign the lowest available gate oxide thickness to the operations needing low leakage units. In this way the performance loss due to high-oxide thickness resources would be compensated by the low-oxide thickness resources. The algorithm attempts to find a suitable time stamp and gate oxide thickness for the off-critical vertices using an exhaustive search. The off-critical vertices are attempted to be placed in each of the control steps within their allowable mobility range. During each placement, gate oxide thickness assignment is done and the CDP value is calculated. The predecessor and successor control steps are adjusted accordingly to maintain the data dependency. A particular vertex is time stamped for a clock cycle with a T_{ox} assignment for which CDP is minimum.

Algorithm 2 Implicit Time Constrained and Resource Constrained Simultaneous Scheduling and Binding Heuristic

Find total number of FUs of all available oxide thicknesses from the DFG : $G(V, E)$.
Get resource constrained as soon as possible schedule C_S.
Get resource constrained as late as possible schedule C_L.
Fix the total number of clock cycles as the maximum of C_S and C_L steps.
Find the vertices in critical path V_c and off-critical path V_{oc} (where, both V_c and $V_{oc} \in V$).
Assume the above C_S schedule as the current schedule S_i.
for all (Critical vertices i.e. $v_i \epsilon V_c$) **do**
    if (Operation in v_i needs high-leaky resources) **then**
        return Assign T_{oxH} resource of type needed by v_i.
    else
        return Assign T_{oxL} resource of type needed by v_i.
    end if
end for
while (All vertices v_i in V are not considered for time stamping) **do**
    if (Vertex v_i needs a high-leaky resource) **then**
        return Assign the highest available thickness T_{oxH}.
    else
        return Assign the lowest available thickness T_{oxL}.
    end if
    Calculate the current delay product of the current schedule CDP_{S_i}.
    for all (Off-critical vertex V_{oc} (i. e. $v \epsilon V_{oc}$) of the current schedule S_i) **do**
        for all (Allowable control step c in the mobility range of v_i) **do**
            Assign next higher thickness if vertex needs high leaky resource and next lower thickness if vertex needs low leaky resource.
            Calculate CDP of the DFG for each control step c.
        end for
    end for
    Fix time stamp of the vertex with the current T_{ox} assignment for which CDP is minimum.
    Remove the above time stamped vertex from list of non-zero mobility vertices.
end while
Calculate gate-oxide leakage power and delay for the scheduled DFG.

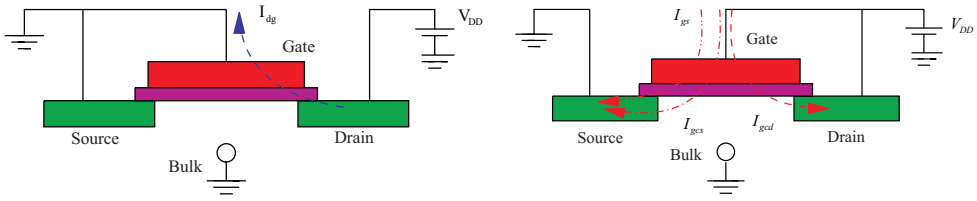
V. Low Gate-Oxide Leakage Datapath Component Library

In this section we describe a methodology for a datapath library creation with different gate-oxide thicknesses in terms of gate leakage and propagation delay for use in behavioral synthesis. This necessitates the development of a seamless design environment that can make the transition between design levels transparent. This design paradigm, using hierarchical components, helps in maintaining modularity as well as making the design space more flexible.

A. Transistor Level

At this stage in the design cycle, we assume that initial silicon is available and BSIM4 predictive CMOS models have been extracted [14]. Fig. 2 shows various components of gate oxide leakage current during the ON and OFF states of an NMOS transistor. The advantage of using SPICE to perform further characterization is that the full dynamic range of the device operation can now be taken into account with very accurate results. The gate-oxide leakage current for a MOS device can be calculated by summing all components:

$$I_{ox\text{MOS}} = I_{gs} + I_{gd} + I_{gcs} + I_{gcd} + I_{gb}. \tag{6}$$

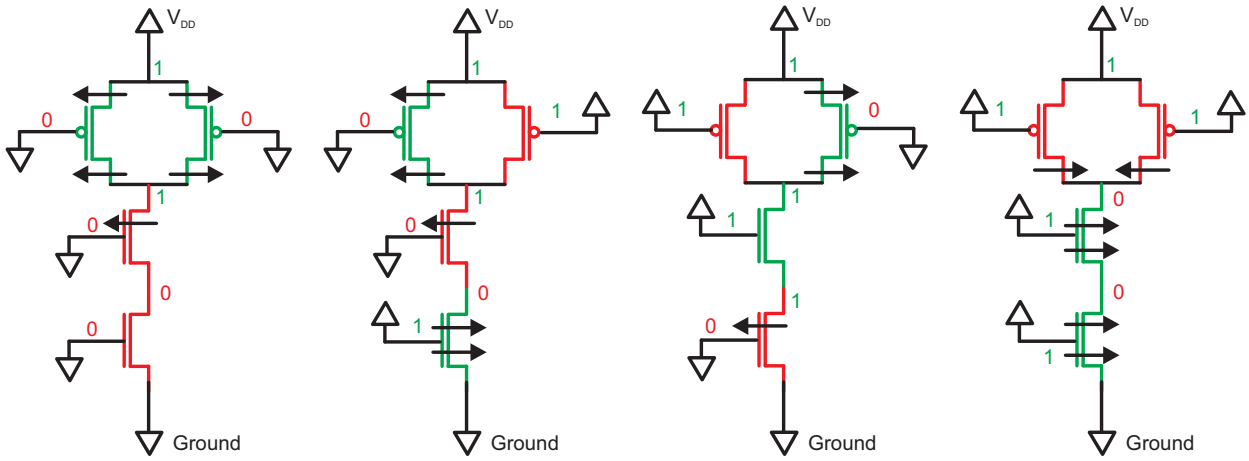


(a) Current in OFF state of NMOS (b) Current in ON state of NMOS

Fig. 2. Gate-oxide tunneling current component flow in the various regions of operation of a NMOS transistor according to the BSIM4 model. I_{gd} and I_{gs} are the components from the diffusions (drain and source, respectively) directly to gate. I_{gcd} and I_{gcs} are the components from the diffusions (drain and source, respectively) to gate via the channel. The gate to bulk tunneling current (I_{gb}) is negligible and not shown.

B. Logic Level

The situation for a logic gate is more complex than a single device since the internal state of the gate and its overall response depend on the values of a number of inputs. We clarify this for a 2-input NAND gate in Fig. 3 which shows the internal states of the transistors as well as the gate-oxide tunneling current paths for all possible combinations of inputs to the logic gate. A similar logic level leakage characterization effort has been presented in [7] and is based on BSIM3 with ad-hoc mechanisms. Since *BSIM3 does not handle gate leakage*, the data presented



(a) input "00" (b) input "01" (c) input "10" (d) input "11"

Fig. 3. Gate-oxide tunneling current paths in various states of a 2-input NAND. V_{Th} voltage drops have not been taken into account.

therein may not be accurate. On the other hand, in this paper we use exclusively BSIM4 models (which, unlike BSIM3 account for gate tunneling) and behavioral simulations; thus our results are more realistic and accurate.

The BSIM 4.4 decks for $45nm$ technology generated represent a hypothetical $45nm$ CMOS process with $T_{ox} = 1.4nm$, $V_{Th} = 0.22V$ for the NMOS and $V_{Th} = -0.22V$ for the PMOS. The nominal power supply is $V_{dd} = 0.7V$. These decks are also scalable with respect to T_{ox} and channel length. The effect of varying oxide thickness was incorporated by varying the parameter $TOXE$ in the SPICE model deck directly. 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 [15], [16]. The PMOS transistors are made twice as wide as the NMOS transistors to ensure proper flow of current through the devices.

The first step in the characterization is the selection of an appropriate capacitive load. A value of 10 times the total gate capacitance C_{gg} of the PMOS device is used [17], [18]. This value depends strongly on the condition of the channel and has been calculated using the device model for each operating condition. The effect of the switching pulse rise time (t_r) is initially examined on the delay characteristics of the various gates. Following standard approaches [19] we define the delay as the time difference between the 50% level of the input and the output waveforms. In order to eliminate an explicit dependence of the algorithm results on t_r, we chose a value that is realistic yet does not affect the delay significantly. For $t_r = 10ps$ the dependence of the delay on t_r is minimal

for all gates and hence t_r can be fixed at that value. The average delay of a logic gate $T_{pd}\text{Logic}$ is calculated as:

$$T_{pd}\text{Logic} = \left(\frac{t_{HL}+t_{LH}}{2}\right), \tag{7}$$

where t_{HL} and t_{LH} are the propagation delay times for high-to-low and low-to-high transitions, respectively.

The gate-oxide leakage current of a logic gate is calculated by evaluating all tunneling components for each PMOS and NMOS device in the logic gate. A total gate-oxide leakage current for the logic gate ($I_{ox}\text{Logic}_{\text{state}}$) for a specific state is then calculated by summing the absolute gate currents over all the MOS devices in the logic gate (both positive and negative gate current contributes to leakage):

$$I_{ox}\text{Logic}_{\text{state}} = \sum_{\forall \text{MOS}_i} |I_{ox}\text{MOS}[i]|, \tag{8}$$

where the index i identifies the device within a logic gate.

During its various states of operation, each logic gate presents different dominant leakage paths, depending on the combination of inputs. For the 2-input gates we considered in this work, the characterization is straightforward as all states can be simulated, thus resulting in a complete characterization. For example, in Fig. 3 we identify the various paths of the tunneling currents in a NAND gate as a function of the inputs. For each of the four possible states (00, 01, 10 and 11), the overall gate-oxide leakage current ($I_{ox}\text{Logic}_{00}$, $I_{ox}\text{Logic}_{01}$, $I_{ox}\text{Logic}_{10}$, and $I_{ox}\text{Logic}_{11}$, respectively) is calculated from Eqns. 6 and 8.

The characterization data for 2-input NAND, NOR, AND, and OR logic gates is presented in Table I. It is clear that the NAND gate outperforms all other logic gates with respect to gate leakage and propagation delay, and hence the NAND realization is beneficial for low gate leakage ultrathin nano-CMOS circuit design.

TABLE I
GATE-OXIDE LEAKAGE CURRENT AND PROPAGATION DELAY FOR VARIOUS LOGIC GATES

Logic Gates	I_{ox} of logic gates in $\left(\frac{nA}{\mu m}\right)$ for different states					T_{pd} in ps
	$I_{ox\,00}$	$I_{ox\,01}$	$I_{ox\,10}$	$I_{ox\,11}$	I_{ox}Average	
NAND2	55.8	172.0	35.8	247.6	127.8	256.9
NOR2	102.1	128.5	121.3	246.6	149.6	378.2
AND2	179.6	295.7	160.0	298.5	233.5	350.0
OR2	225.4	179.6	171.8	297.7	218.6	340.3

C. Functional-Unit Level

In terms of abstraction, this is the highest level characterization in our methodology. From our experimental data in Table I and other existing results [18], [20] it is evident that the NAND realization of functional units is more advantageous in terms of gate-oxide leakage as well as propagation delay. Thus, we propose to realize datapath units using NAND logic gates.

We assume that there are total n_{total} NAND gates in the network of NAND gates constituting an n-bit functional unit out of which n_{cp} number of NAND gates are in the critical path of NAND network constituting the unit. In this model we do not consider the effect of interconnects and focus on the gate-oxide leakage and propagation delay of the functional units only. It may be noted that this assumption does not affect the gate-oxide leakage values as gate-oxide tunneling happens only in the devices, but not in the interconnects.

In the following discussion we will use the symbol p_a to indicate the probability that node a in the functional unit is at logic level "1". Note that the node a is not necessarily an input node and can be an internal node. Similarly, we define $\bar{p}_a = (1 - p_a)$ the probability that the node a is at logic level "0". Even though the discussion assumes two inputs per gate, the theory can be easily generalized for multi-input gates. The average gate-oxide leakage current for a logic gate with inputs a and b is then calculated as:

$$I_{ox}\text{NAND} = \bar{p}_a\bar{p}_b I_{ox}\text{NAND}_{00} + \bar{p}_a p_b I_{ox}\text{NAND}_{01} + p_a\bar{p}_b I_{ox}\text{NAND}_{10} + p_a p_b I_{ox}\text{NAND}_{11}. \quad (9)$$

Once the gate-oxide leakage current per gate is known, we can calculate the total *average* gate-oxide leakage current of an n-bit functional unit in the following manner:

$$I_{ox}\text{FU} = \sum_{j=1}^{n_{total}} I_{ox}\text{NAND}_j, \quad (10)$$

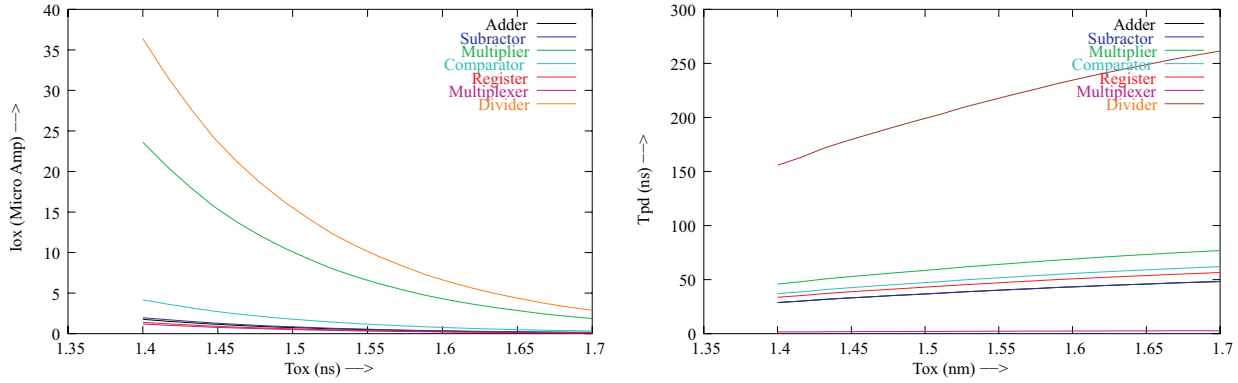
where the index j runs over all the NAND gates in the functional unit. The critical path delay of an n-bit functional unit using the above NAND gates as building blocks is calculated as follows:

$$T_{pd}\text{FU} = \sum_{i=1}^{n_{cp}} T_{pd}\text{NAND}_i. \quad (11)$$

Implicit in Eq. 11 is the assumption that there are as many low-to high transitions as there are high-to-low. Clearly, by probabilistic reasoning this is always the case. In order to use Eq. 10 effectively, the probabilities p_a and p_b must be known. If a and b are primary inputs to the functional unit, then the assumption $p_a = p_b = \frac{1}{2}$

can be used. To address the problem of unknown *a priori* probabilities for the internal nodes of functional units, one approach is to simulate at transistor level the functional unit using the BSIM4 models and SPICE. In order to obtain statistically meaningful probabilities, the simulations must assume random input vectors of very large length. Considering that each gate comprises of several transistors and each functional unit may contain hundreds or thousands of gates, it becomes apparent that such simulations are not expected to complete in a realistic amount of time. We propose, therefore, to estimate the required probabilities via the use of behavioral HDL simulations. The estimation of probabilities will have impact on the estimates of the gate-oxide leakage, however, the efficacy (i.e. percentage reduction) of the proposed algorithms is unaffected.

For the current experiment we characterized a library of 16-bit datapath components, such as adders, subtractors, multipliers, divider, multiplexors and registers following the structural descriptions from [16]. For simulation we considered the case with the probability of both inputs set at equal to 0.5. The results of characterization for $45nm$ technology are presented in Table II. In Fig. 4, we show a comparative view of the variation of the gate-oxide leakage and propagation delay for the components we characterized as the oxide thickness of the transistors changes.



(a) Gate-oxide leakage current Vs oxide thickness

(b) Propagation delay Vs oxide thickness

Fig. 4. Variation of gate-oxide leakage current and propagation delay with respect to gate oxide thickness

VI. Experimental Results

The algorithms presented in this paper were implemented in C and integrated with our in-house behavioral synthesis framework [13]. The algorithms were exhaustively tested with several behavioral level benchmark circuits for several constraints. The resource constraints are expressed as the functional units of different oxide thicknesses and time constraints in terms of performance trade-off factor (T_F). We present the experimental results in this section for selected set of benchmarks and constraints.

TABLE II

DATAPATH COMPONENT LIBRARY FOR VARIOUS GATE OXIDE THICKNESS FOR $45nm$ TECHNOLOGY

Datapath Components	$T_{ox} = 1.4nm$		$T_{ox} = 1.5nm$		$T_{ox} = 1.6nm$		$T_{ox} = 1.7nm$	
	$I_{ox}(\mu A)$	$T_{pd}(ns)$	$I_{ox}(\mu A)$	$T_{pd}(ns)$	$I_{ox}(\mu A)$	$T_{pd}(ns)$	$I_{ox}(\mu A)$	$T_{pd}(ns)$
Adder	1.765620	27.916601	0.686630	36.384698	0.305810	42.291999	0.138480	46.821900
Subtractor	1.973340	27.916601	0.767410	36.384698	0.340430	42.291999	0.155790	46.821900
Multiplier	23.622379	44.484201	9.185840	57.986599	4.131320	67.395603	1.869480	74.622100
Divider	36.397161	151.164796	14.153810	197.036105	6.364310	229.007296	2.885000	253.557994
Comparator	4.189020	35.860901	1.627140	46.744299	0.732790	54.329698	0.328890	60.149698
Register	1.402110	32.679299	0.542380	42.602398	0.242340	49.508801	0.109630	54.824400
Multiplexer	1.194390	1.581100	0.461600	2.066100	0.207720	2.405600	0.092320	2.657800

While calculating the gate-leakage current for single oxide thickness, we used a nominal $1.4nm$ oxide thickness. We considered resources of three dual oxide thickness pairs of (i) $1.4nm - 1.5nm$, (ii) $1.4nm - 1.6nm$, and (iii) $1.4nm - 1.7nm$. For each benchmark and for each pair of dual thicknesses, we present four sets of experiments. In the first set of experiments, we used a smaller number of T_{ox_H} resources and a higher number of T_{ox_L} resources. In the second set of experiments we used a higher number of T_{ox_H} resources as compared to the first set of experiments. In the third set of experiments we used a higher number of T_{ox_H} resources as compared to the second set of experiments. In the fourth set of experiments we relaxed the resource constraints to study the time constrained approach only. The sets of resource constraints are chosen so as to cover functional units consisting of different oxide thickness [9]. They are representatives of various forms of the corresponding RTL representation. The experimental results take into account the tunneling current and propagation delay of functional units and storage units present in the datapath circuit. The percentage reduction in gate-oxide leakage current is calculated as:

$$\Delta I = \left(\frac{I_{ox\,ST} - I_{ox\,MT}}{I_{ox\,ST}}\right) * 100\%. \tag{12}$$

We estimate the critical path delay of the circuit as the sum of the delays of the vertices in the longest path of the DFG. The delay penalty is calculated as,

$$\Delta T_{pd} = \left(\frac{T_{pd_{MT}} - T_{pd_{ST}}}{T_{pd_{ST}}}\right) * 100\%. \tag{13}$$

A. For Algorithm 1

The results for various benchmark circuits for dual thickness technique for $1.4nm - 1.7nm$ are reported in Table III. The reduction in gate-oxide leakage across all the benchmarks ranges from 11% to 76% for different trade-off

factors considered in the experiment and all three dual thickness pairs. We can observe from the results shown that the reduction in gate leakage is in the range of 35% to 85% with a $T_F = 1.5$. We observe that the gate leakage decreases drastically as the number of available T_{ox_H} resources increases. It can be seen that the reduction in gate-oxide leakage is maximum for the DCT and EWF benchmarks, and minimum for the ARF benchmark.

TABLE III
EXPERIMENTAL RESULTS FOR ALGORITHM-1

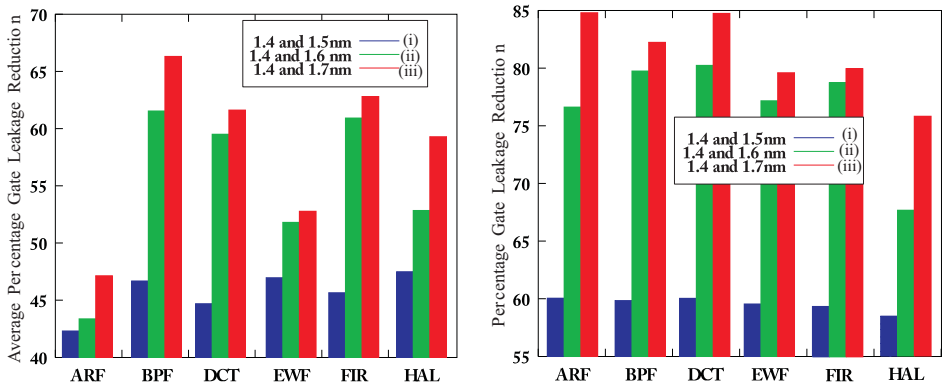
Bench -marks	Res Con	$I_{ox_{ST}}$ (μA)	$T_{CP_{ST}}$ (ns)	Gate-Oxide Tunneling Current ($I_{ox_{DT}}$) in μA and Percentage Reduction (ΔI)											
				T_F=1.0		T_F=1.1		T_F=1.2		T_F=1.3		T_F=1.4		T_F=1.5	
				I_{ox}	ΔI	I_{ox}	ΔI	I_{ox}	ΔI	I_{ox}	ΔI	I_{ox}	ΔI	I_{ox}	ΔI
ARF	1	162.43	831.33	383.30	22.4	379.27	23.3	375.25	24.1	351.10	28.9	343.06	30.6	316.33	36.0
	2	162.43	831.33	317.75	35.7	317.75	35.7	293.60	40.6	269.45	45.5	253.38	48.7	242.72	50.9
	3	162.43	831.33	289.60	41.4	285.57	42.2	281.55	43.0	257.41	47.9	242.72	50.9	225.21	54.4
	∞	162.43	831.33	139.54	71.7	135.51	72.6	131.49	73.4	107.35	78.2	103.32	79.1	75.15	84.8
BPF	1	152.57	689.43	282.03	31.2	278.00	32.2	273.98	33.1	249.84	39.0	245.62	40.1	237.38	42.1
	2	152.57	689.43	141.45	65.5	141.45	65.5	131.30	67.9	113.09	72.4	103.13	74.8	99.11	75.8
	3	152.57	689.43	139.35	66.0	135.32	67.0	113.09	72.4	107.15	73.8	84.72	79.3	77.92	80.9
	∞	152.57	689.43	113.29	72.3	109.27	73.3	85.12	79.2	80.91	80.2	76.89	81.2	72.87	82.2
DCT	1	162.43	823.89	384.04	21.6	380.02	22.4	376.00	23.3	351.85	28.2	347.83	29.0	319.66	34.7
	2	162.43	823.89	187.48	61.7	183.46	62.5	179.44	63.3	155.29	68.3	151.27	69.1	123.10	74.8
	3	162.43	823.89	186.02	62.0	182.00	62.8	177.98	63.6	153.83	68.6	149.81	69.4	121.64	75.1
	∞	162.43	823.89	139.18	71.6	135.16	72.4	131.14	73.2	106.99	78.1	102.97	78.9	74.80	84.7
EWF	1	262.07	557.80	280.66	15.4	272.62	17.8	244.45	26.3	236.41	28.7	208.24	37.2	204.22	38.4
	2	262.07	557.80	216.26	34.8	200.17	39.6	172.00	48.1	163.96	50.5	143.82	56.6	143.82	56.6
	3	262.07	557.80	208.22	37.2	192.11	42.0	167.96	49.3	143.82	56.6	135.79	59.0	131.77	60.2
	∞	262.07	557.80	150.76	54.5	142.72	56.9	114.55	65.4	106.51	67.9	78.34	76.3	74.32	77.6
FIR	1	159.32	500.64	151.50	49.1	147.48	50.4	143.46	51.8	135.42	54.5	131.39	55.8	123.35	58.5
	2	159.32	500.64	133.96	55.0	133.96	55.0	119.31	59.9	111.27	62.6	109.81	63.1	109.81	63.1
	3	159.32	500.64	127.36	57.2	123.33	58.5	109.81	63.1	109.81	63.1	107.25	63.9	99.20	66.6
	∞	159.32	500.64	87.71	70.5	83.69	71.9	79.67	73.2	71.62	75.9	67.60	77.3	59.56	79.9
HAL	1	86.14	329.64	160.25	18.2	160.25	18.2	156.04	20.4	151.83	22.5	151.83	22.5	127.67	34.8
	2	86.14	329.64	90.68	53.7	90.68	53.7	86.46	55.9	82.25	58.0	82.25	58.0	58.10	70.3
	3	86.14	329.64	86.30	55.9	86.30	55.9	82.09	58.1	77.87	60.2	77.87	60.2	537.29	72.6
	∞	86.14	329.64	80.05	59.1	80.05	59.1	75.84	61.3	71.62	63.4	71.62	63.4	47.48	75.7

We also present the results for time-constrained scheduling and binding by relaxing the resource constraints (assuming unlimited resources) in Table III. For the time-constrained approach we observe a reduction in the gate leakage in the range 58% to 60% for dual thickness of $1.4nm$ and $1.5nm$, 67% to 80% for $1.4nm$ and $1.6nm$ thicknesses and 76% to 85% for $1.4nm$ and $1.7nm$ dual thicknesses pair with a T_F of 1.5.

The bar charts in Fig. 5 represent the experimental results for all the thickness pairs under consideration for a tradeoff factor of 1.5. The bar chart in Fig. 5(a) shows the average gate leakage reduction for all the benchmarks in the case of the combination of $1.4nm$ thickness gate oxide CMOS devices with $1.5nm$, $1.6nm$ and $1.7nm$ gate oxide thickness devices, respectively. This shows that, for a particular performance requirement with resource constraint in effect, there is a definite gain in the leakage reduction with increase in the thickness of the gate oxide. In Fig. 5(b) we present a time constrained perspective of the result. Here with all resource constraints relieved we

see an appreciable reduction taking place for all the benchmarks tested. When compared to the resource constrained approach this method shows an average reduction of 15% more across all benchmarks.

We summarize the results from our experiments with various benchmarks and present them in Fig. 6. This provides us with a tool for exploring the design space for determining the desired tradeoff between the conflicting requirements of performance and power. In Fig. 6(a), we see that as the trade-off factor increases the leakage decreases, where average percentage reduction for all resource constraints under consideration is presented. We observed a similar trend even for relaxed resource constraints as shown in Fig. 6(b). The results also specify a region of interest in all benchmarks which can be identified as a knee region. The region to the left of the knee achieves comparatively lower reduction in gate leakage with almost no compromise on performance, while the region to the right of the knee gives a significant reduction with some compromise on performance quantified by the delay tradeoff factor. These curves can be used to identify the optimum design point depending on the requirements of the user.

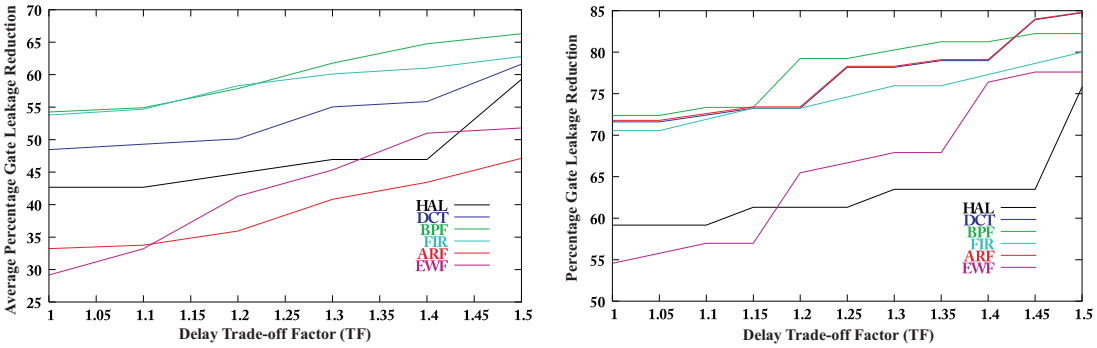


(a) with resource constraints (b) without resource constraints

Fig. 5. Bar charts showing percentage reduction in gate-oxide leakage for three cases for $T_F = 1.5$ using algorithm-1. The W/L ratio of low-oxide NMOS transistor is $180 : 45$ and the W/L ratio of high-oxide transistor are $192 : 48$ for case (i), $208 : 52$ for case (ii), and $224 : 56$ for case (iii). PMOS transistors are twice wider than the NMOS.

B. For Algorithm 2

We carried out our experiments with algorithm 2 using resources of two different gate-oxide thicknesses with similar set of resource constraints as the first algorithm. The results are presented for different constraints for two different oxide thicknesses in Table IV. The quantities with ST subscript represent the values for single oxide thickness and the multiple oxide thickness results are shown with MT subscript. It is observed that while the minimum reduction in the gate leakage current is 44% for the ARF benchmark with resource constraint 1, the



(a) with resource constraints (b) without resource constraints

Fig. 6. Percentage gate-oxide leakage reduction versus performance trade-off factor for design space exploration using algorithm-1.

maximum reduction is 75% for the EWF benchmark with resource constraint 1. The overall reduction for all benchmarks over all constraints is 64% on average. From the results table it can be seen that the reduction in *average* gate-oxide leakage current is maximum for the DCT benchmark, and minimum for the ARF benchmark. As evident from the characterization, the critical path delay increased due to the use of multiple oxide thickness, as delay increases with the increase in oxide thickness. The delay penalty is found to be in the range of $2 - 40\%$ with an overall average of 18%. From the results table it can be seen that the delay penalty is minimum for the EWF benchmark, and maximum for the BPF benchmark.

TABLE IV
EXPERIMENTAL RESULTS FOR ALGORITHM-2

Benchmark Circuits	Resource Constraints	Gate-Oxide Leakage in μA			Critical Path Delay in ns		
		$I_{ox\,ST}$	$I_{ox\,MT}$	% Reduction	$T_{pd\,ST}$	$T_{pd\,MT}$	% Penalty
ARF	1	162.43	66.43	59.10	831.33	978.39	17.69
	2	162.43	47.18	70.95	831.33	1019.96	22.69
	3	162.43	44.83	72.40	831.33	978.39	17.69
	Average I_{ox} Reduction			67.48	Average Penalty		19.36
BPF	1	152.57	61.16	59.91	689.43	863.23	25.21
	2	152.57	49.64	67.46	689.43	831.93	20.67
	3	152.57	44.00	71.16	689.43	863.23	25.21
	Average I_{ox} Reduction			66.18	Average Penalty		23.70
DCT	1	162.43	49.89	69.28	823.89	1033.24	25.41
	2	162.43	59.04	63.65	823.89	1010.83	22.69
	3	162.43	49.89	69.28	823.89	1033.24	25.41
	Average I_{ox} Reduction			67.40	Average Penalty		24.50
EWF	1	262.07	88.68	66.16	557.80	577.76	3.58
	2	262.07	106.29	59.44	557.80	589.26	5.64
	3	262.07	88.68	66.16	557.80	577.77	3.58
	Average I_{ox} Reduction			63.92	Average Penalty		4.27
FIR	1	159.32	51.07	67.94	500.64	576.23	15.10
	2	159.32	55.34	65.26	500.64	512.85	2.44
	3	159.32	51.07	67.94	500.64	576.23	15.10
	Average I_{ox} Reduction			67.05	Average Penalty		10.88
HAL	1	86.14	34.70	59.71	32.44	389.43	18.14
	2	86.14	25.42	70.49	32.44	389.43	18.14
	3	86.14	24.58	71.46	32.44	354.49	7.54
	Average I_{ox} Reduction			67.22	Average Penalty		14.61
For all Benchmarks		Average I_{ox} Reduction		66.54	Average Penalty		16.22

We also carried out experiments using functional units of three different gate oxide thicknesses. In this scenario, for different benchmark circuits the maximum reduction was improved in the range of $3-7\%$ and the average reduction was improved by $2-5\%$. However, there is increase in the average delay penalty for different benchmark circuits, which on an average is $5-11\%$.

C. Comparative Perspective

Since there are no behavioral synthesis research dealing with gate-oxide leakage reduction, direct comparison is not possible. However, in view of the low-power behavioral synthesis works presented in Section II we provide a broader comparative perspective in Table V. In this table ΔP and ΔT_{pd} denote the percentage power reduction and percentage delay penalty, respectively, averaged over all constraints for a particular benchmark circuit. The data are provided wherever available. The work presented in [4] uses a different set of benchmark circuits than the rest of the works in Table V, so we provide the overall average data. The work presented in [5] is area constrained so we did not get the delay penalty data. The results presented show that both dual-T_{ox} algorithms presented in this paper result in significant reduction in gate leakage with reasonable time penalty. They have outperformed the multi-V_{dd} approach for dynamic power reduction and multi-V_{Th} for subthreshold leakage reduction. In summary, we are convinced that multi-T_{ox}, as implemented in the proposed algorithms, is an attractive approach for gate leakage current reduction of nano-CMOS datapath circuits.

TABLE V
A BROAD COMPARATIVE PERSPECTIVE WITH EXISTING LOW-LEAKAGE BEHAVIORAL SYNTHESIS TECHNIQUES

Benchmarks Circuits	Khouri 2002		Gopalakrishnan 2003		This Work (Algorithm 1)		This Work (Algorithm 2)	
	Subthreshold Leakage (Multi-V_{Th})				Gate Leakage (Multi-T_{ox})		Gate Leakage (Multi-T_{ox})	
	ΔP	ΔT_{pd}	ΔP	ΔT_{pd}	ΔI_{ox}	ΔT_{pd}	ΔI_{ox}	ΔT_{pd}
ARF			8.4	–	71.9	22.8	61.7	23.0
DCT	58.0	–			79.8	25.2	68.2	25.1
EWF			19.7	–	77.8	6.0	66.8	5.8
FIR			21.6	–	73.5	12.4	63.0	12.4

VII. CONCLUSIONS AND FUTURE WORK

The gate-oxide leakage due to direct tunneling current is a significant component and contributes to an appreciable portion of total power consumption of nanoscale CMOS circuits. In this paper we presented a technique of simultaneous scheduling and binding of behavioral level elements utilizing functional units of dual oxide thickness. Two algorithms implementing our technique were presented, one with explicit time and one with implicit time

constraint handling. As can be seen from the results of the experiments carried out in this context our dual oxide thickness algorithms are highly effective in reducing the gate leakage with acceptable impact on timing. As gate oxide thickness (T_{ox}) affects threshold voltage (V_{Th}), the proposed methodology can potentially reduce subthreshold leakage, which will be investigated in the future. We have considered the synthesis of datapath circuits, however the work in principle can be extended to control synthesis. The use of multiple dielectrics using high-K dielectric materials along with dual thickness is also being explored for future implementation. The choice of functional units is being implicitly made during scheduling and we are in the process of evaluating its impact on area, capacitance and dynamic power. We are in the process of developing more sophisticated optimization techniques which we believe will further improve the results. Finally, it is our goal in the future to extend the work on tunneling current to a holistic solution to the entire spectrum of power dissipation at the behavioral level. We also aim to account for process variation in the future approaches.

REFERENCES

[1] L. Benini, A. Bogliolo, and G. De Micheli, "A Survey of Design Techniques for System-level Dynamic Power Management," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 8, no. 3, pp. 299–316, June 2000.

[2] Semiconductor Industry Association ITRS, "International Technology Roadmap for Semiconductors," `http://public.itrs.net`.

[3] K. Roy, S. Mukhopadhyay, and H. M. Meimand, "Leakage Current Mechanisms and Leakage Reduction Techniques in Deep-Submicrometer CMOS Circuits," *Proceedings of the IEEE*, vol. 91, no. 2, pp. 305–327, February 2003.

[4] K. S. Khouri and N. K. Jha, "Leakage power analysis and reduction during behavioral synthesis," *IEEE Transactions on VLSI Systems*, vol. 10, no. 6, pp. 876–885, December 2002.

[5] C. Gopalakrishnan and S. Katkoori, "Knapbind: an area-efficient binding algorithm for low-leakage datapaths," in *Proceedings of 21st International Conference on Computer Design*, 2003, pp. 430–435.

[6] X. Tang, H. Zhou, and P. Banerjee, "Leakage power optimization with dual-v_{th} library in high-level synthesis," in *DAC '05: Proceedings of the 42nd annual conference on Design automation*, 2005, pp. 202–207.

[7] A. K. Sultania, D. Sylvester, and S. S. Sapatnekar, "Gate Oxide Leakage and Delay Tradeoffs for Dual-T_{ox} Circuits," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 13, no. 12, pp. 1362–1375, Dec 2005.

[8] N. Sirisantana and K. Roy;, "Low-power Design using Multiple Channel Lengths and Oxide Thicknesses," *IEEE Design & Test of Computers*, vol. 21, no. 1, pp. 56–63, Jan-Feb 2004.

[9] S. P. Mohanty and E. Kougianos, "Modeling and Reduction of Gate Leakage during Behavioral Synthesis of NanoCMOS Circuits," in *Proceedings of the 19th IEEE International Conference on VLSI Design (VLSID)*, 2006, pp. 83–88.

[10] S. P. Mohanty, E. Kougianos, R. Velagapudi, and V. Mukherjee, "Scheduling and Binding for Low Gate Leakage NanoCMOS Datapath Circuit Synthesis," in *Proceedings of the 38th IEEE International Symposium on Circuits and Systems (ISCAS)*, 2006, pp. 5291–5294.

[11] A. Chandrakasan, W. Bowhill, and F. Fox, *Design of High-Performance Microprocessor Circuits*, IEEE Press, 2001.

[12] W. T. Shiue and C. Chakrabarti, "Low-Power Scheduling with Resources Operating at Multiple Voltages," *IEEE Transactions on Circuits and Systems-II : Analog and Digital Signal Processing*, vol. 47, no. 6, pp. 536–543, June 2000.

[13] S. P. Mohanty and N. Ranganathan, "Energy Efficient Datapath Scheduling using Multiple Voltages and Dynamic Clocking," *ACM Transactions on Design Automation of Electronic Systems (TODAES)*, vol. 10, no. 2, pp. 330–353, April 2005.

[14] Y. Cao, T. Sato, D. Sylvester, M. Orshansky, and C. Hu, "New Paradigm of Predictive MOSFET and Interconnect Modeling for Early Circuit Design," in *Proceedings of the IEEE Custom Integrated Circuits Conference*, 2000, pp. 201–204.

[15] Y. Taur, "CMOS Design Near the Limits of Scaling," *IBM Journal on Research and Development*, vol. 46, no. 2/3, pp. 131–139, Mar./May 2002.

[16] N. H. E. Weste and D. Harris, *CMOS VLSI Design : A Circuit and Systems Perspective*, Addison Wesley, 2005.

[17] J. G. Hansen, "Design of CMOS Cell Libraries for Minimal Leakage Currents," M.S. thesis, Dept. of Informatics and Mathematical Modelling, Computer Science and Engineering Technical University of Denmark, Fall, 2004.

[18] V. Mukherjee, S. P. Mohanty, and E. Kougianos, "A Dual Dielectric Approach for Performance Aware Gate Tunneling Reduction in Combinational Circuits," in *Proceedings of the 23rd IEEE International Conference of Computer Design (ICCD)*, 2005, pp. 431–436.

[19] R. J. Baker, *CMOS: Circuit Design, Layout, and Simulation*, Wiley-IEEE, 2005.

[20] K. A. Bowman, L. Wang, X. Tang, and J. D. Meindl, "A Circuit-Level Perspective of the Optimum Gate Oxide Thickness," *IEEE Transactions on Electron Devices*, vol. 48, no. 8, pp. 1800–1810, August 2001.