# A Practical CAD Technique for Reducing Power/ground Noise in DSM Circuits

Arindam Mukherjee
amukherj@uncc.edu

Krishna Reddy Dusety
krdusety@uncc.edu

Rajsaktish Sankaranarayan
rsankara@uncc.edu

Department of Electrical and Computer Engineering
The University of North Carolina at Charlotte, Charlotte, NC 28223, USA.

## ABSTRACT

One of the fundamental problems in Deep Sub Micron (DSM) circuits is Simultaneous Switching Noise (SSN), which causes voltage fluctuations in the circuit power/ground networks. In this work we propose a CAD optimization technique to spread out the switching times of different gates in a circuit to reduce its SSN, by sizing them appropriately. We make sure that its critical delay does not increase while its p/g noise decreases. Our formulation is a Linear Programming one, which we have efficiently formulated and solved. On average, improvements of 28% in the maximum peak-peak voltage fluctuations in the power networks, and that of 20% in the ground networks were achieved by our method over the original circuit implementations. These results were obtained without any performance penalty. As a positive effect of gate-sizing, the power dissipation in the optimized circuits, on average, was reduced to about half of the unoptimized ones for the same supply voltage. We have used standard commercial design flows for all our experiments, and all the results have been validated by extensive SPICE simulations.

## Categories and Subject Descriptors

B.7.1 [**ASIC**] : VLSI circuit designing - *gate sizing to reduce simultaneous switching and power/ground noise.*

## General Terms

Algorithms, Performance, Design, Experimentation.

## Keywords

Simultaneous switching noise, power/ground noise, linear programming, low power, gate sizing, timing analysis.

## 1. INTRODUCTION

### 1.1 Motivation

Deep Sub Micron (DSM) circuits are characterized by high power dissipation and simultaneous switching. High power dissipation is due to fast signal transitions required to achieve high performance, increased circuit densities, and larger leakage currents. High performance implies simultaneous switching, which refers to the condition when two or more logic elements switch at the same time. In that case their time dependent current profiles overlap, which results in high instantaneous currents. These currents are supplied by the power and ground (p/g) networks, whose parasitic resistances and inductances increase with technology scaling. This leads to high values of instantaneous voltage fluctuations of the p/g nodes, which is often referred to as Simultaneous Switching Noise (SSN) or p/g noise.

Operating DSM circuits at low power is often done by scaling the supply voltage down. However, the circuit reliability decreases because the effect of the already substantial p/g noise increases even more. This causes both logic and timing violations. It was observed and analytically proved in [3] that delays of DSM circuits vary with p/g noise by as much as 30% of their typical clock periods. Hence timing closure becomes difficult to achieve. Furthermore, p/g noise can lead to logic failures in dynamic circuits. Thus low power operation of DSM circuits can be achieved by voltage scaling, provided the amount of p/g noise can be reduced in the first place to make the idea feasible. One way to reduce p/g noise is to reduce simultaneous switching, and in this work we propose a Linear Program based gate-sizing methodology to achieve this in DSM circuits.

### 1.2 Previous Work

Recent works have studied the reduction of simultaneous switching. A genetic algorithm based approach [5] was adopted for clock skew optimization to reduce peak current reduction in synchronous circuits. The current drawn by a logic block was modeled as in [2], and it was assumed that the current was independent of the block output loading, as well as its input vectors. It was claimed that maximum peak current reduction can be as much as 50% without penalty on cycle time and average power dissipation. A graph based scheduling algorithm was similarly used to reduce p/g noise by optimizing clock skew in [4]. In this case, the authors achieved an average reduction of 19.6% in the peak current, and an average reduction of 38.7% in the current swing. However the benchmarks of the above papers were not actually laid out on silicon, and in the absence of p/g networks, the effect of their technique on p/g noise could not be determined. Besides, the above works do not discuss the difficulties involved in actually synthesizing clock trees to achieve the desired clock schedules.

A linear programming (LP) based gate-sizing and buffer insertion method was proposed in [1] to ensure that only one input of any multi-input gate switches at any given time. The idea was to reduce glitches at the gate outputs under zero wire delay model, so that transient energy could be minimized. The peak transient energy was reduced by 47%, and the average energy was reduced by 27% in [1] for a specific circuit. However the authors assumed that any


*GLSVLSI'03*, April 28-29, 2003, Washington, DC, USA.

and all gate delays, as found after the LP optimization, could be realized by a physical gate in a given technology.

In the next section we shall state our problem formulation, followed by a discussion of the delay characterization of our technology specific gate library in section 3. In section 4 we present our linear programming formulation to reduce simultaneous switching by gate-sizing. The experiments and results are then described in section 5, and finally in section 6 we conclude with a brief discussion of our current and future research in this area.

## 2. PROBLEM FORMULATION

Given a netlist which has been mapped to a library of gates, we propose sizing the gates, so that simultaneous switchings at the outputs of different gates can be minimized. Note that differently sized gates of the same type will have different delays while driving a certain output load. Each gate in the library has different sizes, and each sized gate needs to be characterized for delay under different output capacitive loading conditions.

## 3. LIBRARY CHARACTERIZATION

We have used a commercial standard cell CMOS library in the 0.18 μm technology as our base library. For each basic gate G in the library, we derived different sizes by resizing the different transistors in the gate, such that the output rise and fall times remain same. Note that all gates have single outputs. The resized gates were named from G_1 to G_N, where N is a number between 5 and 80. For any particular resized version G_s of G, MOSIS 0.18 μm BSIM level 53 SPICE models were used to model the gate along with the related parasitics. The output of G_s was then loaded with capacitances varying from 0 fF till 150 fF in steps of 2-5 fF, and SPICE simulations were done to gather data on the output delay dependence of the gate on the capacitive loading. The dependence of delay on capacitive loading was found to be largely linear.

The above process was repeated for all library gates and all sizes, and thus the expanded library was extensively characterized for delay. We shall now formally define some of the concepts introduced in this section.

***Definition 1*** : Base library is the original commercially available standard cell library.

***Definition 2*** : A basic gate G is a component of the base library. A sized gate G_s with the size code s, is a resized version of G.

***Definition 3*** : Expanded library is the super set of all sized versions of all basic gates in the base library.

***Definition 4*** : A node in a netlist corresponds to a basic gate after the netlist has been mapped to the base library.

The upper limit of output capacitive loading for the delay characterization was found in the following manner. Starting from VHDL descriptions of our benchmark circuits, we mapped them to the base library, and laid them out using a commercial tool flow. These layouts were then extracted for parasitics, and the wiring parasitics at the different node outputs were converted into their effective output capacitances. Thereafter, the SPICE netlists of the circuits were derived by back-annotation. This gave us the effective output wiring capacitances of all the nodes in the netlist, the maximum value of which was less than 90fF. So we decided to characterize our expanded library gates for a maximum output loading of 150fF. This was done to account for the sink capacitances of a driver gate, which will further load its output.

Thus for any basic gate G, we have a 3-tuple {s, C, δ(G.s.C)}, where s is the size code of G, C is its output load capacitance used for delay characterization, and δ(G.s.C) is the average of its output rise and fall times as found from simulations. δ(G.s.C) is uniquely identified by the parameters G, s and C.

## 4. LINEAR PROGRAMMING APPROACH

### 4.1 LP variables

In order to minimize simultaneous switching in DSM circuits, we formulate our gate-sizing problem as a linear programming (LP) one. A gate is represented by a node in a circuit graph, where the nets form the edges. Each node has a corresponding 'd' variable and a 't' variable associated with it. The 'd' variable is the delay of the node, while the 't' variable represents the arrival time of the latest input of the node.

### 4.2 LP constraints

***Procedure 1 :*** In section 3 we described how the effective output wiring capacitances of all the nodes in a netlist are estimated. For any node $n_m$, we find its effective output capacitance by adding its estimated output wiring capacitance with the capacitances of its fanouts. Note that the fanout nodes of $n_m$ are known from the netlist. We assume that the fanout capacitances correspond to the inputs of the respective minimum sized gates that are available in the expanded library. This can be justified because most of the gates chosen are minimum sized. Thus we can estimate the effective output capacitance $CO_m$ of $n_m$.

Let G be the corresponding basic gate of the node $n_m$. Considering the 3-tuple {s, C, δ(G.s.C)} of G, we find the maximum value of δ(G.s.C), given by $\delta(G.C)_{max}$, for which the value of C matches $CO_m$ most closely. Then the upper bound of the gate delay is :

$$d_m \leq \delta(G \cdot C)_{max}, \forall m \quad (1)$$

Similarly, the lower bound of the delay of $n_m$ is given by the minimum value of δ(G.s.C) for which the value of C matches $CO_m$ most closely. This is denoted by $\delta(G.C)_{min}$. Thus,

$$d_m \geq \delta(G \cdot C)_{min}, \forall m \quad (2)$$

The arrival time of an input c to the node $n_m$ is given by:

$$a(c)_m = d_f + t_f \quad (3)$$

where f identifies the node $n_f$, which is a fanin node of $n_m$. The latest input arrival time at $n_m$ is then found as:

$$t_m = maximum(a(c)_m), \forall c \quad (4)$$

which means that $t_m$ is the maximum input arrival time, among all inputs to $n_m$. For the LP formulation, equation (4) can be written out using several equations like $t_m \geq a(c)_m$, where the number of such equations will be equal to the number of inputs of $n_m$. That is, c will iterate over the different inputs of $n_m$. Using equation (3), $t_m$ can be expressed as:

$$t_m \geq d_f + t_f, \forall c \in n_m \quad (5)$$

Here f identifies the node $n_f$, which is the source of the c input of node $n_m$.

This equation does not provide an upper bound of $t_m$ however. The latter is found as follows. The switching time of node $n_m$ is then given by $t_m + d_m$. In order to ensure that the critical path delay does not exceed a certain maximum value $D_{crit}$, we impose the

additional constraints :

$$t_m + d_m \leq D_{crit}, \forall n_m \in PO \quad (6)$$

where PO is the set of all nodes whose outputs are primary outputs of a circuit. Let us now introduce some additional concepts.

***Procedure 2 :*** We levelize our circuits, so that each node $n_m$ has an associated level whose value is an integer denoted by $L(n_m)$. The level of any node is found as follows. All primary inputs have level 0, while for a node $n_m$, $L(n_m)$ is 1 plus the maximum of its input level values.

***Definition 5*** : For any circuit, its *ckt-depth* is the maximum value of the node levels over all nodes in the circuit.

***Definition 6*** : The fanout set of a node $n_m$, $FO(n_m)$, is the set of nodes driven by the output of $n_m$. The transitive fanout of a node $n_m$, $TFO(n_m)$, is the set of all nodes that can be reached from $n_m$. This includes $FO(n_m)$, and recursively includes the nodes in the fanout sets of $FO(n_m)$ and so on, till nodes in PO (equation (6)) are encountered.

***Definition 7*** : The criticality $\alpha(n_m)$ of a node $n_m$, is the maximum value of the node levels over all nodes in $TFO(n_m)$.

$$\alpha(n_m) = maximum[L(n_i)], \forall n_i \in TFO(n_m) \quad (7)$$

A high criticality value of $n_m$ implies that more critical paths pass through $n_m$. If $\alpha(n_m)$ equals the *ckt-depth*, $n_m$ exists on the most critical path of the circuit. Note that our basic assumption here is that the delay of a path is primarily determined by the number of levels of nodes that it passes through, rather than on the functionalities of the nodes.

***Definition 8*** : The fanout-score $\beta(n_m)$ of a node $n_m$, is the total number of nodes in $TFO(n_m)$, considering reconvergent fanouts.

## 4.3 LP cost function

For the node $n_m$, we define a slack-factor $\Gamma(n_m)$ as:

$$\Gamma(n_m) = [w\alpha(n_m) + \beta(n_m)]^{-1} \quad (8)$$

where w is a weighting coefficient. The higher the value of w, the more weight is assigned to the criticality of a node compared to its fanout-score. Note that the slack-factor of a node decreases if the node is on more critical paths, or if the node transitively fanouts to a lot of other nodes in the circuit. The intuition behind this model is that if a node $n_m$ is on one of the more critical paths, or if the node fanouts to a lot of other nodes, delaying $n_m$ to reduce its chance of simultaneous switching with some other nodes, would have a high probability of increasing the circuit delay.

In procedure 2 we described our circuit levelization method. For any particular level $\lambda$, we select those nodes $\{n_i\}$ having $L(n_i) = \lambda$. All nodes in the set $\{n_i\}$ are sorted based on their respective slack-factors in a descending order into a list $U_\lambda$. From this list, we then select the first node $n_j$, and the last node $n_k$. The first node has the highest slack-factor, while the last node will have the lowest one in $U_\lambda$. Thus $n_j$ can be delayed more than $n_k$, which means that the output switching time of $n_j$ ($t_j + d_j$) can be made more than that of $n_k$ ($t_k + d_k$). The upper bound on the switching delay of $n_j$ is imposed by equation (6). In order to reduce simultaneous switching, we try to maximize (($t_j + d_j$) - ($t_k + d_k$)). Then the nodes $n_j$ and $n_k$ are removed from $U_\lambda$, and the above process is repeated for a different node pair till $U_\lambda$ is empty, or there is just one node in it. This is done till all the levels in the circuit have been considered. Note that we simplify our cost function by separating out the switching times of node pairs in the same level, and do not consider doing the same for two nodes which exist in different levels. The reason is that nodes in different levels do indeed have naturally separated switching times because of node propagation delays. However, nodes in the same level have the tendency of simultaneously switching because in high performance circuits, all inputs to nodes at a certain level are required to arrive within a narrow window of time. Thus, our cost function is:

$$\sum_{\lambda=1}^{ckt-depth} \sum_{n_j, n_k \in U_\lambda} [(t_j + d_j) - (t_k + d_k)] \quad (9)$$

This is a linear cost function as required for the LP formulation. All our constraints in equations (1) through (6) are linear too.

In our LP formulation, the cost function of equation (9) is maximized subject to the constraints (1), (2), (5) and (6). Note that our optimization is library dependent because the constraints (3) through (6) are dependent on the library.

## 4.4 Gate-sizing

The output of our optimization are certain delay values that the nodes would be required to have, in order to minimize simultaneous switching. Let a node $n_m$ have the delay value *dv*. We shall then find the 3-tuple corresponding to the gate type G of that node, whose C has the closest match to the estimated output capacitance $CO_m$ of the node, and whose $\delta$(G.s.C) has the closest match with *dv*. In that case, s will be the size code for that node. Note that $CO_m$ for $n_m$ can be determined by using procedure 1. It was experimentally observed that the distribution of the gate sizes chosen by our method, was skewed to the range between 1 and 20.

# 5. EXPERIMENTS AND RESULTS

## 5.1 Experimental set-up

Starting from .blif descriptions of our benchmark MCNC circuits, we converted them into VHDL models using our own software, and optimized them using the commercial tool flow provided by Mentor Graphics. The optimized structural VHDL models were then mapped to the 0.18 µm base library which is available in the flow. The circuits were then laid out, and parasitic extraction was done. The power and ground (p/g) networks were laid out as meshes using the same tool flow, which inserts decoupling capacitors. Thereafter the SPICE netlists of the circuits, with parasitic extraction and distributed modeling, were derived by back-annotation. These included the package parasitics, as well as coupling capacitors. The netlists were then simulated using a commercial SPICE simulator, and their p/g noise, critical delays and RMS power dissipations were measured. We refer to these results as "original". Note that the critical delay of a circuit as measured in this way, is used as the $D_{crit}$ value in equation (6).

As part of our optimization (section 4.4), we have gate-sized the original optimized structural VHDL netlists using sizes from the expanded library. A value of 10 was assigned to the weighting factor w in equation (8) because that gave us the best results during optimization. We refer to the optimized circuits as "our" in the following result tables. These circuits were also laid out like the

"original" ones, extracted for parasitics, and their back-annotated SPICE netlists were simulated to measure p/g noise, critical delays and RMS power.

## 5.2 Results

**Table 1: Power mesh voltage (mV) comparison**

| Circuit | Maximum Peak-Peak | | Absolute Maximum | | Absolute Minimum | |
|---|---|---|---|---|---|---|
| | Original | Our | Original | Our | Original | Our |
| 9sym | 99.48 | 20.79 | 1828 | 1816 | 1710 | 1733 |
| cordic | 96.41 | 76.98 | 1824 | 1807 | 1704 | 1709 |
| decod | 52.54 | 39.95 | 1811 | 1805 | 1748 | 1760 |
| c17 | 8.87 | 3.46 | 1802 | 1800 | 1793 | 1796 |
| cm138a | 28.62 | 10.77 | 1805 | 1801 | 1775 | 1789 |
| alu2 | 194.87 | 148.27 | 1870 | 1842 | 1605 | 1638 |
| exmpl2 | 226.23 | 208.01 | 1845 | 1824 | 1602 | 1625 |
| Average | 101.00 | 72.60 | 1826 | 1814 | 1705 | 1721 |

In table 1 we compare the power mesh voltages in the original and our gate-sized circuits. The 10x10 power meshes for the circuits were supplied by a single pad at a chip corner. The local power distribution for logic inside a mesh square is done using power trees, rooted at the mesh node. Column 1 shows our MCNC benchmarks. The maximum peak-peak voltage at a power node is the difference between the maximum and minimum voltages at the power node, over thousands of random input vectors. In the second and third columns we have compared the maximum peak-peak voltages, over all power mesh nodes, for the original and optimized circuits. The greatest value of the maximum voltage at any power node, over thousands of random input vectors, is the absolute maximum voltage of the node. In the following two columns we have compared the absolute maximum power mesh voltages, over all power mesh nodes, in the original and the optimized circuits. Similarly, the absolute minimum power node voltages of the circuits, over all power mesh nodes, have been compared in the next two columns for the original and optimized circuits. All these parameters are metrics of p/g noise because they are voltage fluctuations of the power nodes from their ideal value of 1.8 volt (or 1800 mV) in the 0.18 μm technology.

From the table we can conclude that our method consistently achieves lower p/g noise than the original circuits, for all the benchmarks and for all the p/g noise metrics. On average, our method achieves 28% reduction in the maximum peak-peak power mesh voltage, compared to that of the unoptimized circuits.

Similarly, in table 2 we depict the p/g noise reductions achieved by our methodology in the ground meshes of the circuits. Like the power meshes, the 10x10 ground meshes were powered by a single pad from a chip corner. On average, our method achieves 20% reduction in the maximum peak-peak ground mesh voltage, compared to that of the unoptimized circuits.

The RMS power dissipations of the circuits were significantly impacted by our gate-sizing method. For lack of space, we are not tabulating the experimental data here. However we noticed that on average, the mean RMS power can be reduced to about 1/2 of that dissipated by the original circuits. We have also compared the critical path delays of the different circuits designed using our approach, with those of their original implementations. We found that on average, our circuit implementations do not have any delay penalty vis-a-vis the original circuits. Comparing the circuit areas in general, we observed that on average, there was no increase in area over those of the original layouts.

**Table 2: Ground mesh voltage (mV) comparison**

| Circuit | Maximum Peak-Peak | | Absolute Maximum | | Absolute Minimum | |
|---|---|---|---|---|---|---|
| | Original | Our | Original | Our | Original | Our |
| 9sym | 97.06 | 85.75 | 97.16 | 73.44 | -2.81 | -1.73 |
| cordic | 107.82 | 89.17 | 107.78 | 100.04 | -1.63 | -0.43 |
| decod | 60.35 | 34.43 | 60.27 | 33.89 | -1.31 | -0.97 |
| c17 | 8.80 | 4.00 | 8.80 | 3.95 | -1.38 | -0.37 |
| cm138a | 30.53 | 13.61 | 30.50 | 13.61 | -1.20 | -0.35 |
| alu2 | 182.14 | 149.79 | 182.35 | 161.08 | -2.06 | -1.13 |
| exmpl2 | 234.98 | 202.11 | 234.90 | 220.86 | -0.95 | -0.24 |
| Average | 102.81 | 82.69 | 103.11 | 86.70 | -1.62 | -0.75 |

# 6. CONCLUSIONS AND FUTURE WORK

In this work we have demonstrated that by gate-sizing we can effectively reduce simultaneous switching, and hence, reduce p/g noise. This leads to low power and reliable operations of DSM circuits. We have efficiently formulated and solved our gate-sizing problem as a Linear Programming one. On average, we achieve improvements of 28% in the maximum peak-peak voltage fluctuations in the power networks, and that of 20% in the ground networks compared to those in the original circuit implementations. With about 2 times reduction in power dissipation, and in the absence of any speed penalty compared to the original circuits, the results look extremely promising.

Our current and future work is focussed on using clock skew optimization, with gate-sizing and buffer insertion, to achieve even better results. The delay of a gate depends on its input vector and slew rates, as well as on its size and output loading. This leads to the gate having not one, but a range of delay values (timing window) during circuit operation. We also plan to consider these timing windows in our future optimization.

# REFERENCES


[1] V.D. Agrawal, M.L. Bushnell, G. Parthasarathy, and R. Ramadoss, "Digital Circuit Design for Minimum Transient Energy and a Linear Programming Method", International Conference on VLSI Design, India, January 1999. pp.434-439.

[2] A. Bogliolo, L. Benini, G. D. Micheli, and B. Ricco, "Gate-level current waveform simulation of CMOS integrated circuits", Proc. of ISLPED,1996. pp.109-112.

[3] L. -H. Chen, M. Marek-Sadowska and F. Brewer, "Coping with buffer delay change due to power and ground noise", Proceedings of Design Automation Conference, 2002, pp.860-865.

[4] W-C.D. Lam, C-K. Koh, and C-W.A. Tsao, "Power Supply Noise Suppression via Clock Skew Scheduling", Proc. of ISQED, March 2002. pp.355-360.

[5] P. Vuillod, L. Benini, A. Bogliolo, and G. De Micheli, "Clock-skew optimization for peak current reduction," Kluwer Journal of VLSI Signal Processing, vol. 16, no. 2-3, 1997. pp. 117-130.