Gate Sizing Under Uncertainty

Nathaniel A. Conos([✉]), Saro Meguerdichian, and Miodrag Potkonjak

University of California, Los Angeles, 4403 Boelter Hall,
Los Angeles, CA 90095, USA
{conos,saro,miodrag}@cs.ucla.edu

Abstract. We present a gate sizing approach to efficiently utilize gate switching activity (SA) and gate input vector control leakage (IVC) uncertainty factors in the objective function in order enable more efficient power and speed yield trade-offs. Our algorithm conducts iterative gate freezing and unlocking with cut-based search for the most beneficial gate sizes under delay constraints. In an iterative flow, we interchangeably conduct gate sizing and IVC refinement to adapt to new circuit configurations. We evaluate our approach on benchmarks in 45 nm technology and demonstrate up to 62 % (29 % avg.) energy savings compared to a traditional objective function that does not consider SA and IVC. We further adapt our approach to optimize yield objectives by addressing processing variation (PV). Significant improvements were achieved under identical timing yield targets of up to 84 % max (55 % avg.) and 74 % max (25 % avg.) mean-power savings for selected ISCAS-85 and ITC-99 benchmarks, respectively.

Keywords: Gate sizing · Low power · Input vector control · Switching activity · Yield optimization

1 Introduction

Gate sizing is a powerful optimization technique used to minimize power and/or area under strict timing constraints by altering the widths of transistors in gates. Gate sizing has been extensively studied over the past three decades [2–5] and several approaches have been proposed. Previous approaches, however, do not consider optimization uncertainty factors, such as switching activity (SA) and the impact of input vector control leakage (IVC), which greatly impact the overall optimization strategy. Additionally, the impact technological uncertainty, i.e., process variation (PV), has increased in the deep-micron regime, and traditional techniques lack the ability to effectively address yield targets. As a result, the modern design flow imposes a number of modeling and optimization challenges, that require new methods in accounting for uncertainty, both technological and optimization.

One major challenge is the simplification of timing and power models, which may lead to suboptimal solutions when mapping out to real designs, thus, increasing optimization uncertainty. Accounting for accurate gate and interconnect delay and its dependencies on capacitive load slew are often ignored [5].

A. Orailoglu et al. (Eds.): VLSI-SoC 2013, IFIP AICT 461, pp. 23–47, 2015.
DOI: 10.1007/978-3-319-23799-2_2

Additionally, nominal gate switching activity and/or average gate leakage are generally assumed in previous works limiting the potential improvements by accounting for realistic operating conditions. Moreover, previous approaches are either dynamic or leakage power-centric in their optimization flows, which do not address the varying application usage characteristics present in high-performance systems (e.g., data-centers, super-computing) to energy constrained mobile devices (e.g., tablets, smart-phones).

Cell library-based optimization has emerged as the de facto standard for modeling power and delay of a circuit design. Many previous approaches, however, utilize simplified timing models by assuming convex and/or linear delay and power models [4]. Empirical analysis has shown that accurate timing models are non-linear/non-convex. Furthermore, optimizing circuit designs using a discrete cell library, however, leads to solving an NP-Hard problem [6]. As a result, many heuristics have been developed in order to address the huge optimization problem search space. A major drawback of these methods is that they require heavy parameter tuning and are difficult to reproduce, since they are technology dependent and are driven by a set of sensitivity functions. These methods often perform iterative per-gate or per-group improvement and are too compute intensive and are impractical to be applied on modern IC sizes, even with incremental updates. Furthermore, these approaches mainly perform local optimization (i.e., local-moves) and are susceptible to be trapped in local minimas [7]. In this work, we interchangeably use the term gate and cell to represent the granularity in which we size gates, which is at the cell-level.

The usage of modern cell libraries, however, have enabled the support of various supply/threshold voltages, and drive strengths, thus, enabling a rich performance and energy trade-off to address the potentially vastly differing device usage characteristics. However, current tools do not account for realistic conditions into their objective functions (e.g., gate activity, duty cycle, input vector control), with respect to their applications, potentially impacting obtained results.

In this chapter we focus on two main contributions. The first contribution, introduced in [1], improves state-of-the-art sizing methodologies by simultaneously considering gate switching activity (SA) and gate input vector control (IVC) uncertainty. The key contribution is that significant benefits of incorporating actual gate SA and gate IVC in the objective function over the equivalent approach that only uses averaged values. Additionally, we show how the obtained solution varies when accounting varying duty cycles. The second contribution, introduced in this chapter, is the integration our SA+IVC technique with a pre-characterization step to improve power and yield targets.

The focal point of our approach is a scalable gate sizing algorithm that considers gate SA and IVC leakage. The steps are to: (1) extract the SA of gates based on simulation of real workloads; and (2) conduct IVC to obtain the input vector that induces the lowest total leakage energy across all gates, and (3) an iterative gate sizing approach freezes maximally-constrained gates (ones that are at high-power states as determined by SA and IVC) while searching for a sizing option that best improves the current picture. The objective function in step 3

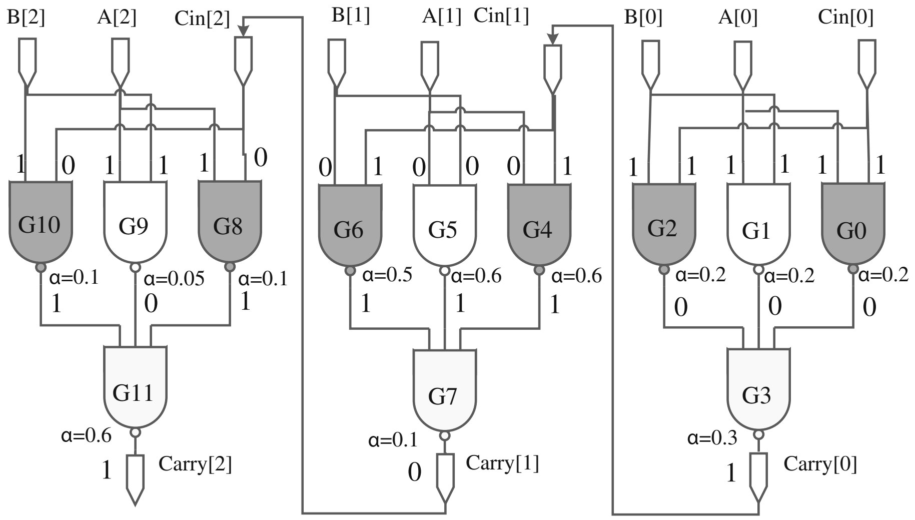


Fig. 1. Carry propagation for 3-bit carry-ripple adder

to be considered at the iteration depends on the types of options available and their impacts on both delay and energy. The algorithm prevents the algorithm from reaching a local minima by freezing gates as they are sized until all gates have been frozen, then unfreezes all gates, re-conducts IVC (since new gates may be energy-dominant), and reiterates steps 2 and 3 until the solution converges or the delay constraint cannot be met.

The concept introduced in this chapter is the extension of our gate sizing technique to account for the impact of process variation for maximizing yield targets with respect to specified delay and power targets. Efficient yield optimization is achieved by a pre-characterization step which identifies the most critical cells that are likely to impact delay targets through an efficient Monte Carlo simulation that considers epsilon-paths (ε-paths). The intuition is to simultaneously speed-up critical paths in order to minimize the impact of process variation across generated circuit instances and minimize the power costs by conducted gradual gate sizing.

We evaluate our gate sizing approach on benchmarks included in ISCAS-85/89, ITC99 and arithmetic units first without consideration of PV. Our results indicate over 62 % (29 % avg.) energy improvement over a method that assumes nominal SA and IVC, demonstrating that gate SA and IVC play an major role in the guiding sizing decisions over an equivelent sizing algorithm that does not. We then present results using our PV-aware technique, which further imroves our original gate sizing algorithm to address yield objectives. We demonstrate the effectiveness of our process variation-aware (PV-aware) against a non PV-aware technique across ISCAS-85 and ITC99 benchmarks, achieving 64 % and 48 % power savings with respect to identical target delays.

2 Motivation

We begin by providing a small realistic example demonstrating the importance of considering both SA and IVC uncertainty factors in the gate sizing optimization process. Consider the carry propagation of a 3-bit carry-ripple adder, shown in Fig. 1. Assume that 2- and 3-input NAND gates have input-dependent leakage power consumption values for two possible sizes, small (X1) and large (X2), shown in Table 1. Also assume that the given input vectors ($A = 101, B = 101$, and $C_{in} = 1$) are realized throughout the entire duration of the application. Figure 1 shows the input vectors to each gate. Therefore, the overall leakage power of the circuit is 288 nW. For simplicity, ignoring load and slew dependencies, assume that all gates have delay of 10 ps at size X1 and 5 ps at size X2. Finally, assume that at the beginning of the optimization process, all gates are nominally sized to size X1. Therefore, there are eight nominal critical paths (colored), $\{G0, G2\} \rightarrow G3 \rightarrow \{G4, G6\} \rightarrow G7 \rightarrow \{G8, G10\} \rightarrow G11$, with nominal delay 60 ps. Consequently, total leakage energy of the circuit is 1.73×10^{-17} J.

As an example, consider a delay constraint of 55 ps. It is clear that one of gates G3, G7, or G11 should be sized up to X2, as all critical paths pass through these bottleneck gates and decrease the delay of each of these gates will decrease the overall delay. A traditional approach to gate sizing would consider these gates equally. In other words, increasing the size of either would decrease delay and increase switching and leakage power by the same amounts. However, from Table 1, we see that the leakage power of a gate, due to transistor stacking, strongly depends on its applied inputs, with up to a 43X difference between the lowest-leakage state (input vector "100": 1.29 nW) and highest-leakage state (input vector "111": 55.8 nW) of a 3-input NAND gate. Furthermore, switching energy of a gate is directly proportional to its activity factor, or the likelihood that the gate will switch. Therefore, because the gates have both different applied input vectors and different activity factors, sizing up each one will have a different effect on overall power and energy consumption, so they should not be weighted equally in the optimization process.

First, consider the case where the duty cycle of the adder is low and therefore leakage energy dominates. We can determine from Table 1 that increasing the size of gates G3, G7, or G11 will increase leakage power by 9.96 nW, 167.42 nW, or 56.35 nW, respectively, while decreasing the overall delay by 5 ps. Therefore, the optimal decision is to increase the size of gate G3, which will have minimal impact on leakage energy, increasing leakage power to 298 nW and *decreasing* leakage energy to 1.64×10^{-17} J. Increasing the size of G7 would instead increase leakage power to 455 nW, *increasing* leakage energy to 2.50×10^{-17} J. Thus, considering IVC in this example in the optimization algorithm can improve the energy by roughly 60 %.

Now, consider the high duty cycle scenario, where switching energy is the dominant factor. Again, for simplicity, assume that all gates consume 10 nJ and 20 nJ of switching energy at nominal activity factor 1.0 for a given application at sizes X1 and X2, respectively. Figure 1 shows the activity factors (α) for each gate. Therefore, overall switching energy consumption at the nominal size is 35.5 nJ. In this case, increasing the size of gate G7 is the optimal decision, since it has the lowest activity factor and consumes less switching energy than when up-sizing either G3 or G11. In fact, this decision results in a switching energy of 36.5 nJ, whereas increasing the size of G11 would result in a switching energy of 41.5 nJ. Therefore, the decision that considers SA performs roughly 14 % better.

Another key component modern design flow that requires attention is the process variation (PV) impact on sizing moves. For example, sizing alterations, such as the ones discussed in this section, are mostly performed in the pre-silicon phase. Therefore, there is uncertainty in how sizing alterations are affected by on-chip variations once the design has been manufactured. Optimizing for yield can be exhaustive and require extensive statistical characterization; however, conducting a pre-characterization step may reveal more suitable circuit alterations that may not be possible to capture in a traditional static flow. For example, designers may leverage spatial and temporal correlations in a design to determine the best cell(s) to size. For example, G3 may belong to a path whose delay is more susceptible by PV; thus, it would be more beneficial for G3 to be up sized eve though G7 was identified to be a more efficient move in terms of power-speed trade-off under a traditional static optimization approach. As a result, the benefits of accounting the impact of PV in a circuit is clear when optimizing for yield.

Table 1. NAND gate leakage values (nW) for two sizes (X1, X2) based on input vector control (IVC) from a single threshold 45 nm cell library [8], where min and max leakage states are represented by bold and italicized fonts, respectively.

	NAND-3	
IVC	X1	X2
000	3.32	13.28
001	18.18	72.73
010	4.21	16.84
011	39.49	157.97
100	**1.29**	**5.15**
101	18.78	75.13
110	3.76	15.04
111	*55.8*	*223.22*

	NAND-2	
IVC	X1	X2
00	**3.48**	**13.93**
01	24.8	99.2
10	4.09	16.34
11	*37.21*	*148.83*

To present these motivations, we have made a number of assumptions that when relaxed make the optimization much more complex in practice. It is reasonable to assume that additional information (gate switching, input vector state, and pre-characterization statistics) can be readily obtained by modern CAD tools and/or by implementing a simple gate-level simulator combined with statistical packages. Such information is beneficial since it enables the simultaneous consideration of low duty cycle and high duty cycle scenarios, as in real use cases at current and pending deep-submicron feature sizes, leakage and switching may both have significant impacts on overall energy. For example, sizing up G7 in the high duty cycle scenario may in reality not be optimal, since its input gates have higher values for α than, the input gates of G3, and thus their switching energies would increase by larger factors. Thus, this IVC depends on how the circuit is sized and its duty-cycle. Therefore, a feedback loop exists between gate sizing and IVC that must be addressed simultaneously during the optimization. The example here demonstrates that both IVC and SA are crucial considerations in gate sizing for energy optimization in the presence of delay deadlines. Furthermore, combining PV-uncertainty can further provide key insight in guiding circuit alterations.

3 Related Work

We cover a set of related gate sizing approaches that have considered a variant of SA or IVC. Several approaches exist that address continuous and discrete gate sizing. Common methods to solve the gate sizing problem have been convex optimization [4], Lagrangian Relaxation [2,3,17], and gradient and sensitivity-based optimization [9,18].

Gate sizing methods have also been combined with V_{dd} and V_{th} assignment to minimize power under various gate SA ratios [10,11]. These works, however, have only considered average leakage values when accounting for leakage and have not explored real application activity factors when considering gate switching activity. Leakage minimization using IVC is a popular technique for due to its strong dependency on the input vector state [12]. IVC and gate replacement techniques have also been combined [13] by replacing gates at their worse-case leakage state with equivalent gates with lower leakage power.

To the best of our knowledge, we are the first to consider gate sizing in the presence of both SA, IVC, and duty cycle. Prior approaches have at most considered one or two terms accurately [16], and/or do not differentiate between the duty cycle with respect to switching and leakage energy weights, leaving many approaches to be either dynamic or leakage power-centric. For example, the state-of-the-art gate sizing contest considers only nominal leakage power [5]. Our technique minimizes total energy, such that both the switching and leakage energy components are accurately accounted for in accordance to their usage or duty cycle.

4 Cell Library Energy and Delay

The total energy of a CMOS integrated circuit can be characterized into two main components: (1) dynamic (switching) energy due to charging of input pin/output load capacitance's; and (2) static (leakage) energy, which we model from the dominant sub-threshold leakage and gate leakage currents. Thus, the total energy consumed can be computed as:

$$E_{total} = E_{switch} + E_{leak} \tag{1}$$

$$E_{switch} = \sum_{i}^{N} es(g_i), \qquad E_{leak} = \sum_{i}^{N} el(g_i) \tag{2}$$

es and el represent the switching and leakage energies, respectively, for gate g_i. es is the product between probability that a gate's input pin j will switch, α (SA), and the estimated full-cycle (e_{fc}) power consumed from propagating a signal from input pin j to output pin k. el is the sum of leakage energies consumed at each possible leakage state of a gate, which is also dependent the ratio of the total time spent at each leakage state for both *active* and *standby* (idle) periods. The total time (T) is directly proportional to product of the circuit delay (D) and total cycles, where D represents the critical output-pin arrival time ($rise$ or $fall$) of a primary output gate $ot^{r,f}(g_i)$:

$$D = max(ot^{r,f}(g_i)) \qquad s.t.\ g_i \in G_{out} \tag{3}$$

G_{out} represents a circuit's set of primary output gates. Therefore, the delay of a circuit can be determined by solving:

$$\begin{aligned} ot^{r,f}(g_i) = dl^{r,f}(g_i) + max(ot(fin_j^{f,r})) \\ s.t.\ fin_i \in FI_i \end{aligned} \tag{4}$$

$fin_j^{f,r}$ is the $fall, rise$ arrival time of a fan-in gate j in the set FI_i of gate g_i. Note that the propagation of delay depends on the unateness assumption. For simplicity, we assume all cells are negative unate, thus, rise (r) and fall (f) gate delays are propagated as assumed to the next stage.

We use a cell table library look-up as [5] to model gate *rise* and *fall* delay ($dl^{r,f}$) as a function of its input slew (transition time), and driving load. However, we use an alternate 45 nm cell library (Nandgate) [8] to account for switching

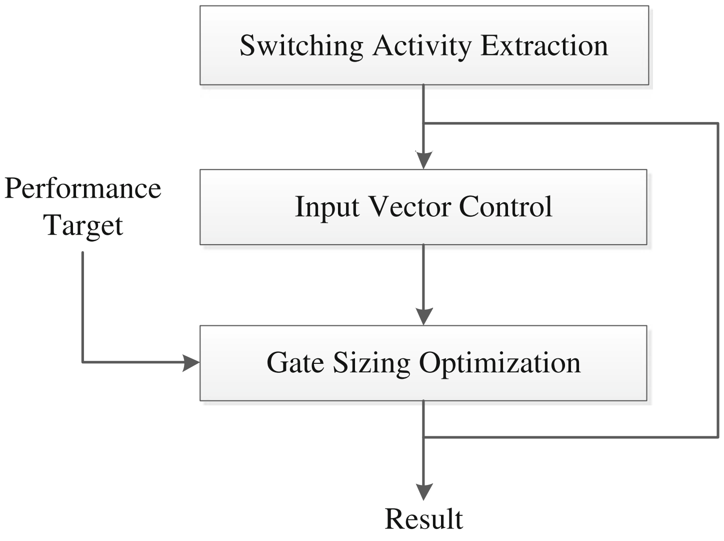


Fig. 2. Gate sizing optimization flow

and input vector dependent leakage power, which are obtained in a similar look-up table fashion, provided per-input pin accurate switching, and input vector state probabilities, which can be obtained using gate-level simulation.

5 Technical Approach

5.1 Gate Sizing

Our gate sizing procedure is composed of three major phases (Fig. 2). The first phase extracts gate switching activity factors (SA) for a given circuit by performing event-driven gate simulation from a set of input bit vectors. Figure 3 illustrates an example SA extraction for a carry-look-ahead unit (cla4) from two applications (mpeg2enc/dec). The second step identifies a primary input bit vector that places gates in low leakage states in order to minimize the total energy of the circuit, which accounts for leakage consumption for both active (obtained from SA) and idle periods. IVC techniques range from random simulation to satisfiability (SAT) and model counting-based formulations. The final component is the gate sizing algorithm, where the goal is to minimize total energy consumption under a delay constraint. The approach is iterative; at each iteration, gates are either frozen or unlocked based on their leakage (IVC) and switching (SA) impact, while a search is conducted for the most beneficial current *move*.

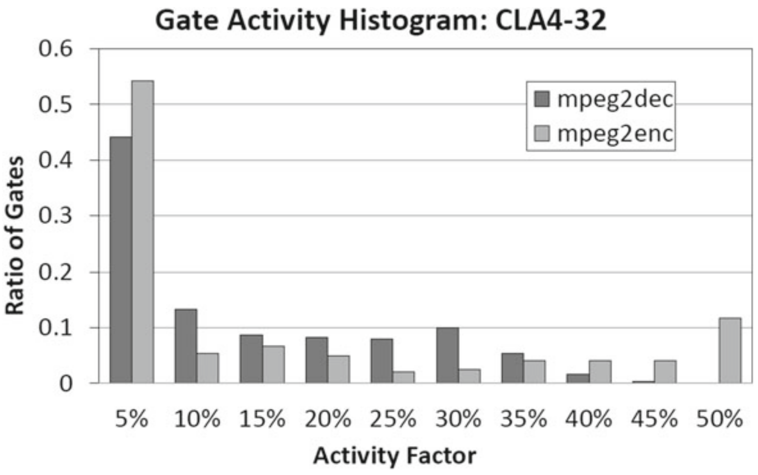


Fig. 3. Gate switching activity for a 32-bit CLA circuit when using real mpeg2enc/dec application input stimulus. Shown are varying distribution of gate activity within a circuit and across two applications.

Our algorithm is sensitivity-based in nature in terms of determining which move or set of moves to perform. A gate sizing move can have 1 of 3 effects (increase, decrease, have no effect) on 2 parameters (energy and delay), leading to a total of 9 separate possible classes for a move. The algorithm classifies each move to and enforces a priority scheme in order to select a move that has higher precedence. There are three precedence levels, where level 1 is the highest priority. Moves that improve both parameters are at precedence 1, moves that improve just one parameter and do not affect the other are at precedence 2, and moves that improve one parameter at the expense of degrading another are of precedence 3. Note that moves that degrade both parameters are never selected. Each precedence level has its own objective function for selecting the best move: (1) the product of the respective improvements; (2) the single improvement; and (3) the normalized ratio of improvement and degradation.

The algorithm considers a cut of M gates at a time and restricts one gate to be sized per group visit. Once a size move is committed, the gate is locked and is no longer considered within that phase. The completion of a phase is defined as having locked all gates, or having no more acceptable moves among sizable gates that improve the objective function. The algorithm terminates after the solution converges or if a target delay (D_{target}) can not be met after a number of phases. All gates are unlocked before the start of each new sizing phase.

The algorithm initially freezes the top K energy-critical gates by setting them to their minimal sizes at the beginning of the phase. We note that this initial set potentially restricts some delay critical gates, as improving the delay of these gates may be required in meeting a deadline. To relax this constraint (i.e., if a solution cannot be obtained), K is relaxed through a locking threshold ratio γ, where a new K is computed (e.g., $K = K^{'} \cdot \gamma$), thereby, enabling potentially more delay critical gates to be reduced. It is crucial to identify the top K energy-critical gate, which in turn depends on both SA and IVC; this maximally-constrained gate locking is one of the key innovations of the approach, and prevents being trapped at a local minima by encouraging global circuit optimization.

We utilize an epsilon tree to minimize circuit delay updates (ε_{path}), which consists of gates that were on the critical path during the last accurate delay computation (Fig. 4a). Since the critical path may change during optimization, we also include the immediate fan-out gates of each critical gate (e.g., nodes 1 and 3), fan-in nodes may be added for greater accuracy as their slews may also impact timing propagation. The figure shows bold-outlined nodes (e.g., 7, 8, 9, 5 and 6) are the primary outputs (G_{out}), and are transitively connected to at least one node belonging to the critical path (e.g., 0, 2, 4, 8). Thus, the delay cost of sizing a gate on the ε_{path} can be estimated by the sum of its δ_i with respect to the target delay (D_{target}) is used to estimate the delay impact of each move via a delay cost formula, as shown below:

$$D_{cost} = \sum_{i}^{|G_{out}|} (\delta_i - D_{target})^2 \tag{5}$$

This formulation enables very efficient delay estimation by only considering the delay impact of a small subset of gates at a time. A drawback of this approach, however, is that a potentially new critical path may emerge. This remains to be a major challenge for existing gate sizing techniques that attempt to maintain delay accuracy during optimization [9,17,18]. To address this issue, the frequency of delay updates can be increased by adjusting M and γ to be larger values, as we have done. These parameters can be adjusted, to trade-off accuracy vs run-time.

Our used values of M and γ achieved a delay accuracy to be within 5 %, while achieving linear run-time scaling with respect to circuit size (Fig. 4b).

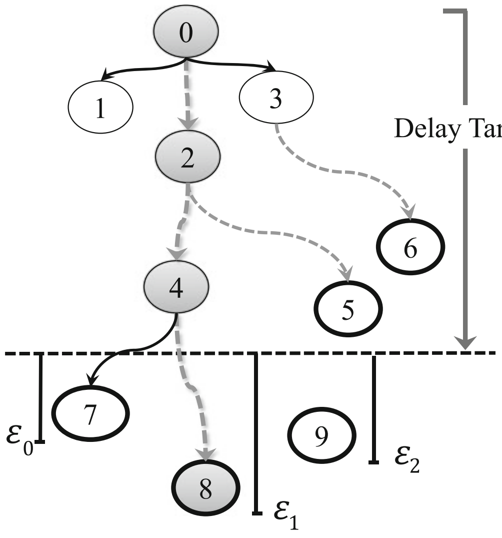
5.2 Maximizing Yield

In this section, we extend our gate sizing technique to account for the impact of process variation in order to maximize yield targets with respect to delay and power.

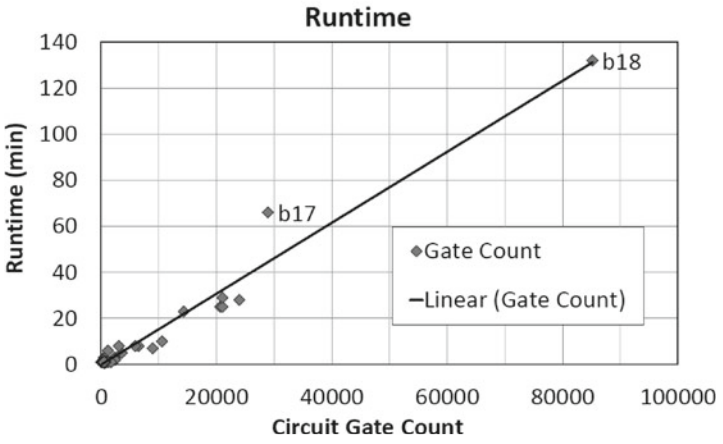
Cell Characterization. As a pre-processing step, we characterize each cell's ability to improve the circuit delay when its size is increased by one. The objective is to capture its delay impact on a set of ε-critical paths when its size is increased. We define the ε-critical paths as the set of paths whose cell slack is within an ε-threshold value with respect to the critical slack. The distinguishing factor of our PV gate sizing method against our non-PV method is that we aim to improve a set of ε-critical paths. Doing so improves the circuit's resiliency against the potentially harmful effects of PV by over-optimizing one or few critical paths that do not consider PV.

The major challenge of our cell characterization phase is that it incurs additional run-time overhead of $O(n^2)$, since the delay impact of sizing each cell is required to be computed. The timing overhead, however, can be minimized by considering a smaller ε-critical set and is practical, since this step can be performed only once or a few times. Additionally, the typical distribution of critical cells in the circuit is anywhere from 1 % to 10 % during the initial stages of our gate sizing method when all cells are initially set to their valid minimal size configuration. It is important to note, however, the number of cells that belong within ε-critical paths increases as the circuit becomes more sensitive. At these inflection points, additional characterization phases can be performed in order to improve accuracy.

Figure 5 presents the circuit slack impact (y-axis) when increasing the size of each cell (x-axis) independently. The y-axis represents a normalized slack sum value across a set of 1000 generated circuit instances with process variation using 3-sigma normal threshold-voltage and gate effective length distribution [19]. The x-axis lists cell id's ranked in ascending order from left to right with respect to its *initial* slack value for a circuit without PV. It is important to note that a larger slack for a given cell indicates that it has a greater potential to improve circuit delay (increase overall slack) when it is up sized. Also note that cells are originally listed in ascending order with respect to their original slack computed without PV, thus demonstrating that greedily up sizing the most critical gates (left-most gates) does not yield a result in the most delay improvement when increasing their size. In fact, some cells that are not on the critical path yield better delay improvements than cells with smaller slack, as shown with some cells in between 60 and 160 for circuit $c432$. This behavior can be attributed to cells that may not be initially on the critical path, but are an immediate fan-out of a cell that is, and therefore, increasing their size may cause subsequent upsizing of their critical transitive fan-in cells, thus improving circuit delay greater



(a)



(b)

Fig. 4. (a) An example of ε critical path (ε_{path}); the critical path in dashed-red; transitive fan-out output nodes in bold outlines; and ε_i corresponds to the absolute delay difference w.r.t to the target delay used for estimating delay cost of a move. (b) The linear run-time of the new gate sizing approach

than individually sizing a gate on the critical path. For our approach, we then rank the cells by their normalized slack value when considering which cells are more critical with respect to timing when conducting gate sizing.

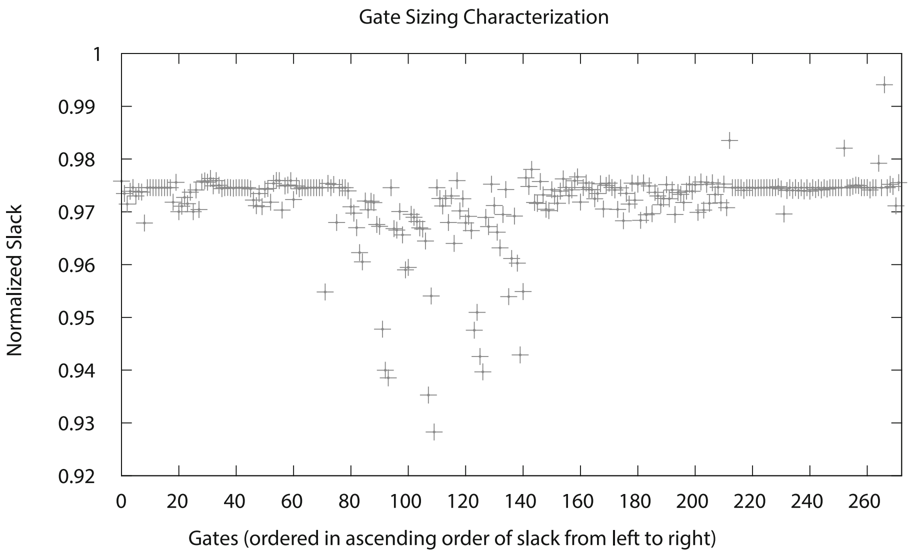


Fig. 5. Impact of independent gate sizing. Gates are ordered left to right in ascending order with respect to their slack (without PV). Larger normalized sum slack values (y-axis) indicate greater ability to improve delay (reduce delay, increase slack) across 1000 generated circuit instances with PV when size is increased to the next discrete size.

PV-Aware Gate Sizing Algorithm. We present several extensions to our PV-aware gate sizing approach for yield optimization. Specifically, these extensions are introduced to guide which cells are chosen to be sized first based on their pre-processing characterization result described in the previous subsection. As previously mentioned, the cells which are first considered to be sized are based on their normalized slack sum value acquired during the pre-processing characterization step and are ranked in descending order. Thus, the cells with the largest slack-sum are sized or configured first.

We conduct the same cut-based search gate sizing algorithm presented in Sect. 5.1. However, we also further prune the potential candidate cells to be sized by sizing cells that fall under an ε-critical path, as well as further selecting cells which are the least constraining cells (smallest size) to be sized. We define the least constraining cells as the set of cells currently in the sizing phase set that are currently configured as the minimum size among the cells belonging to the ε-critical paths. For example, assume that the current ε-set include cells A, B, C, and D with sizes 1X, 1X, 2X, and 3X, respectively. Then, cells A and B are to be considered to be sized first during this phase. To break ties between cells A and B, their respective cost score (Eq. 5) is multiplied by their respective normalized slack sum value. Increasing the size of these cells enable more balanced ε-critical paths, thus, effectively enabling more efficient delay-power trade-offs to be performed in later sizing iterations that achieve tighter delay constraints; this is achieved since optimizing a set of ε-critical prevents from over optimizing a few paths, which may cause certain paths to converge to a timing wall.

6 Experimental Results

We evaluated our gate sizing approach on a set of benchmarks in ISCAS-85/89, ITC-99 suites, as well as integer arithmetic units consisting of adders (carry ripple, carry-look-ahead, Kogge-Stone) and multipliers (array, Dadda). All units were synthesized using a single threshold (HVt) 45 nm open cell library from [8] under the typical cell configuration. An in-house timing/power engine was implemented in C++ and was correlated to an industrial tool, Synopsys PrimeTime, to be within 10^{-3}ps. All results were optimized using identical rules such as ensuring no slew or load violations exists in the final design, as presented in [5]. The only differences in our framework is the choice in cell library, which was done in order to enable accurate IVC computations, as well as the choice of circuit benchmarks. In handling slew and load violations, we adopt an iterative approach as proposed in [18].

The SA of gates and IVC for each circuit were obtained from simulation of random input vectors. However, real application switching activity factors were obtained from mpeg2enc/dec benchmarks from recorded operand values from each unit type, running ARM7TDMI-ISA mpeg2-enc/dec traces [14,15]. The initial simulation parameters set were, $K = 25\,\%$, $M =$ twice the length of average critical path, $\gamma = 0.2$, and were fixed across all benchmarks. The delay target for each circuit was set as the median between the achieved delay when all gates were set to their maximal size, and the achieved delay when all gates are at their minimal size. Five duty cycle scenarios (D0 = 10 %, D1 = 20 %, D2 = 33 %, D3 = 50 %, and D4 = 100 %) were considered.

6.1 Gate Sizing Under Switching Activity and IVC Uncertainty

We first evaluate our sizing algorithm under two gate sizing assumptions: (1) SA+IVC, which considers gate switching activity factors and input vector control in the objective function; and (2) *Base*, where the objective function uses only nominal gate switching (50 %) and average gate leakage values for total energy computation. Table 2 compares the two methods, where Max (%) savings corresponds to the maximum energy improvement achieved over the *Base* method across the five duty cycle cases (D0 to D4) for each circuit under the same timing constraint. As expected, the maximum improvement observed varies across duty cycles and circuits, motivating the advantage of utilizing accurate power and delay knowledge.

Table 4 provides overall energy improvements across the benchmark suites. The results generated by the new approach achieved a maximum energy improvement of 62 % for circuit c2670 and 29 % average overall for the same delay.

Figure 6 provides a normalized energy and delay plot for c2670, which illustrates the advantage of using more accurate power and delay information. A delay of 0.87 shows that the *Perceived* energy deviates from the trend of the *Actual*

Table 2. Energy savings when considering gate *SA* and *IVC* during the gate sizing procedure over the *Base* method. The obtained switching and leakage energies are presented for the *SA+IVC*. The maximum energy deltas (Δ %), corresponds to the max difference in energy profile "perceived" by the *Base* method during optimization.

Circuit	No. gates	Max energy savings			$SA + IVC$					
		Total (%)	Sw (%)	Lk (%)	Sw (μJ)	Max (Δ%)	Lk (μJ)	Max (Δ%)	Duty Cycle	Delay (ns)
c880	383	8.14	8.16	8.05	125.11	31.22	15.75	3.01	D2	0.73
c1355	554	14.35	15.01	13.26	264.13	31.12	164.79	15.19	D1	0.74
c1908	932	13.41	13.72	9.41	405.7	29.57	32.06	6.75	D4	1.14
c7552	3568	43.65	44.02	41.32	1685	30.27	284.6	3.21	D2	1.20
c5315	2330	58.72	59.1	54.57	855.6	32.93	87.58	3.19	D4	1.34
c432	272	30.15	35.24	22.35	68.54	33.15	53.55	23.67	D1	0.62
c2670	1202	62.64	62.74	60.78	407.6	31.91	24.62	1.39	D4	0.85
c3540	1703	28.84	29.66	24.14	799.1	31.09	152.2	2.47	D2	0.79
s1488	698	46.38	46.49	44.81	182.6	35.54	14.09	12.05	D2	0.42
s1494	692	35.5	36.08	33.84	283.9	36.36	103.3	12.55	D1	0.42
s15850	10547	31.34	30.76	34.48	3803	33.69	662.1	4.98	D3	2.22
s838	473	32.49	38.89	27.17	139.3	31.76	199.8	10.84	D1	1.64
s5378	3054	16.76	30.08	15.06	826.6	36.2	7876	26.34	D0	0.68
s9234	5897	50.19	50.52	47.23	2041	32.51	242.5	4.43	D3	1.50
s38417	23963	53.2	53.85	46.17	6661	30.65	714.2	8.58	D3	1.29
s35932	21035	22.27	22.31	21.25	8317	29.28	332.1	37.26	D3	0.80
s38584	18161	29.01	31.52	28.75	5820	32.24	59720	4.77	D0	1.52
b10	204	44.69	45.06	32.43	40.56	34.45	1.55	54.83	D2	0.36
b11	633	35.53	35.79	31.74	250.9	32.37	18.77	61.52	D2	1.10
b12	1183	22.91	27.89	3.31	314.0	37.03	107.3	60.54	D1	0.61
b13	375	22.05	22.78	15.01	161.23	31.85	18.4	55.33	D1	0.33
b14	6498	28.42	28.65	26.21	3024	33.22	320.7	54.94	D2	1.31
b15	8920	27.62	27.7	26.75	1666	38.99	166.1	54.24	D3	1.58
b17	28911	21.25	25.29	20.79	8.53	38.76	80.58	55.99	D3	1.68
b18	85188	7.02	8.67	5.96	44.54	37.84	71.4	54.72	D1	2.10
b20	14322	27.85	28.57	24.07	3.66	33.2	0.74	55.8	D2	2.08
cra32	225	38.32	45.14	37.76	6.72	44.90	92.22	32.09	D0	2.08
cla432	305	22.25	25.02	12.18	10.18	42.87	3.28	13.62	D2	0.35
ks32	611	24.2	23.18	24.63	17.64	44.76	40.83	14.6	D0	0.30
arr8	512	35.96	36.3	23.18	178.5	34.69	22.84	21.29	D1	0.81
dad8	542	30.35	30.99	36.33	101.3	35.83	9.12	11.98	D2	0.62

energy plot. In performing move-trace analysis, we noted that the *Base* method caused the algorithm to over-size a few selected critical paths and encountered a timing wall much earlier, whereas *SA+IVC* was able to efficiently trade-off delay for an additional 0.05 delay units, as shown.

Table 3. Energy savings of ($SA{+}IVC$) over $Base$ using extracted gate switching activity and input vector control from mpeg2enc/dec applications assuming a (D2) "33 %" duty cycle. The units represent an single-adder (32b) and multiplier (8b) configuration of an ARM7TDMI core [15].

mpeg2	Energy savings			cla432				dad8			
	Total (%)	Sw (%)	Lk (%)	Sw (μJ)	Sw Imp (%)	Lk (μJ)	Lk Imp (%)	Sw (μJ)	Sw Imp (%)	Lk (μJ)	Lk Imp (%)
enc	15.31	17.15	8.55	742.7	4.73	628.7	1.09	2267	20.61	498.6	13.92
dec	25.10	29.89	6.21	11.02	6.42	24.30	1.20	101.4	30.99	9.24	21.58

Figure 7 shows a cumulative distribution of gate switching and leakage energies of the max improved result for $SA{+}IVC$ over $Base$ for circuit c2670. Our approach shows that accurate knowledge enabled the algorithm to efficiently guide the circuit to a lower energy state, as shown with higher percentage of gates falling under lower energy profiles for both leakage and switching energy. This is important to note since due to the difficulty of comparing gate sizing algorithms, many existing algorithms are sensitivity-based in nature, thus, the ability to guide an algorithm to determine more promising "moves" greatly impacts the optimization procedure.

Table 3 presents results comparing the minimal configuration found by $SA{+}IVC$ and the perceived minimal configuration obtained by $Base$. The minimum energy configuration determined was cla432 and dad8, optimized under the same timing constraints determined by the multiplier. For these configurations,

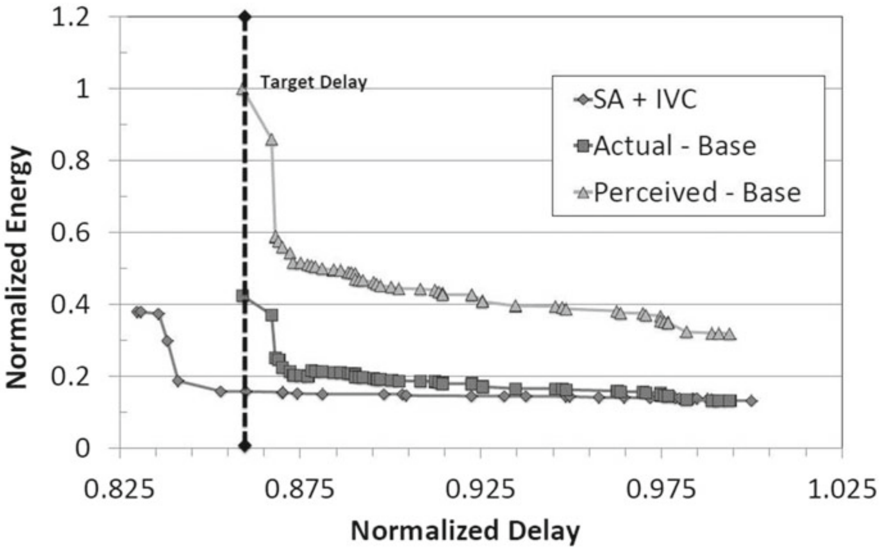


Fig. 6. Energy vs delay plot of c2670. The $SA{+}IVC$ approach consistently outperforms the $Base$ method.

Table 4. Overall energy savings with respect to benchmark suite

Benchmark suite	Max tot (%)	Avg. tot (%)	Avg. Sw (%)	Avg. Lk (%)
ISCAS-85	62.64	29.70	30.58	26.74
ISCAS-89	53.20	28.33	29.99	26.96
ITC-99	58.83	29.23	30.90	24.15
Arith	57.19	30.48	33.23	33.15

our approach shows additional savings in both leakage and switching categories where the majority of the savings for both cases (15 % mpeg2enc, 25 % mpeg2dec) were achieved by the multiplier circuit.

6.2 Yield Optimization

Figure 8(a–b) compares the power reduction (%) of using a PV-aware gate sizing approach against one that does not. Each result compares the highest performance target (smallest delay), where both the PV-aware and non-PV-aware solution are able to satisfy. Average power savings of up to 64 % and 48 % were achieved for ISCAS-85 and ITC-99 benchmarks, respectively. We compare the highest comparable performance delay target since solutions obtained at lower performance delays require few circuit alterations, thus, achieve similar power results.

The complete sizing optimization procedure can be observed in Figs. 9 and 10 for ISCAS-85 and ITC-99 benchmarks, respectively. Two sizing solutions are

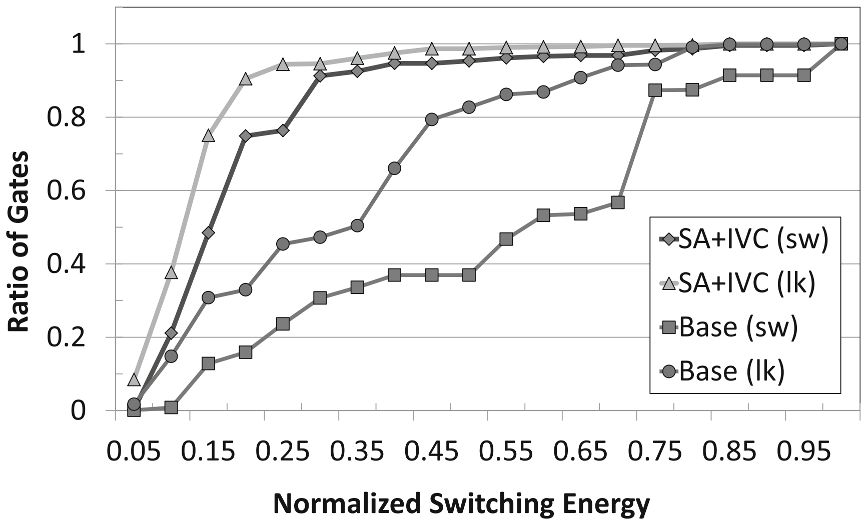
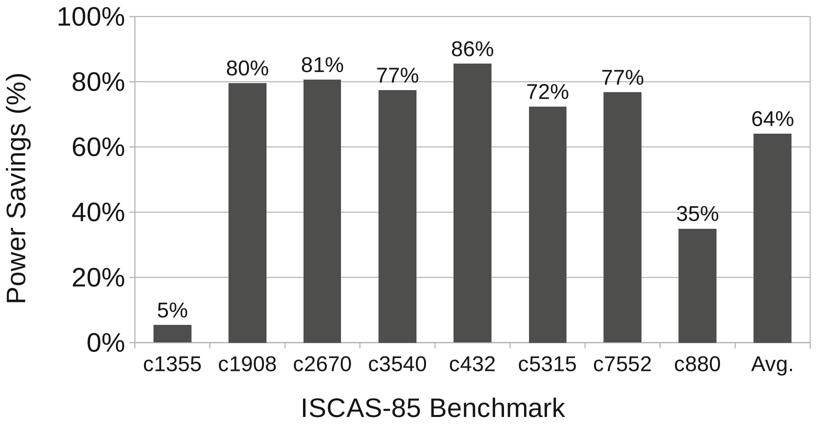
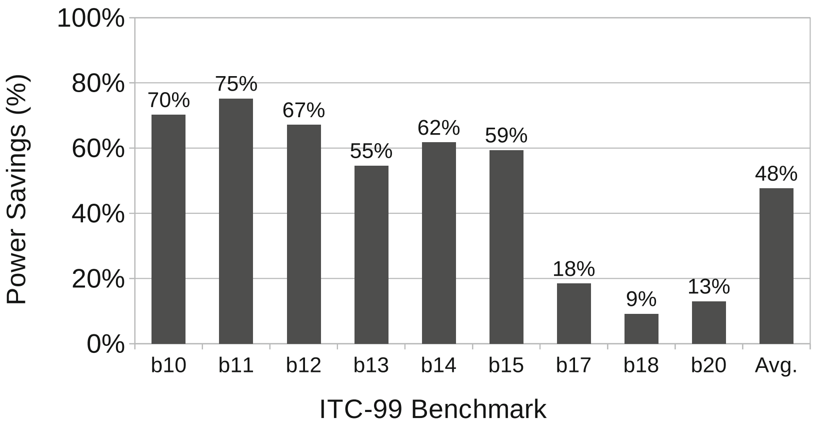


Fig. 7. Cumulative distribution of leakage and switching energy after sizing for gates in c2670. The accurate *SA+IVC* approach results in a higher percentage of gates at lower energy.



(a)



(b)

Fig. 8. Power savings (%) achieved when comparing generated sizing solutions using a process variation-aware over a non-process variation-aware gate sizing technique. Reported savings are generated from the minimum delay result achieved with respect to each approach.

compared for each benchmark: (1) PV-aware gate sizing (red circle); and (2) non-PV gate sizing (blue triangle). Each point corresponds to a sizing phase where up to K cells are sized simultaneously. Therefore, for each circuit, the gate sizing procedure begins with all cells set to their minimal power configuration and is represented by the right-most point. The left-most point corresponds to the fastest achieved delay for each sizing procedure.

Significant savings were achieved in six out of the eight ISCAS-85 benchmarks since the enhanced PV-aware algorithm was able to avoid hitting a timing wall.

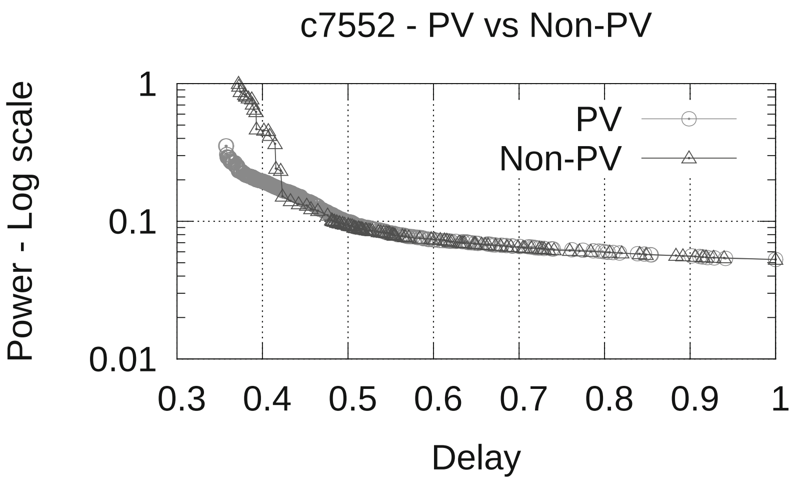


Fig. 9. Selected ISCAS-85 gate sizing result comparing the process variation-aware (PV) and non-process variation-aware (non-PV) techniques

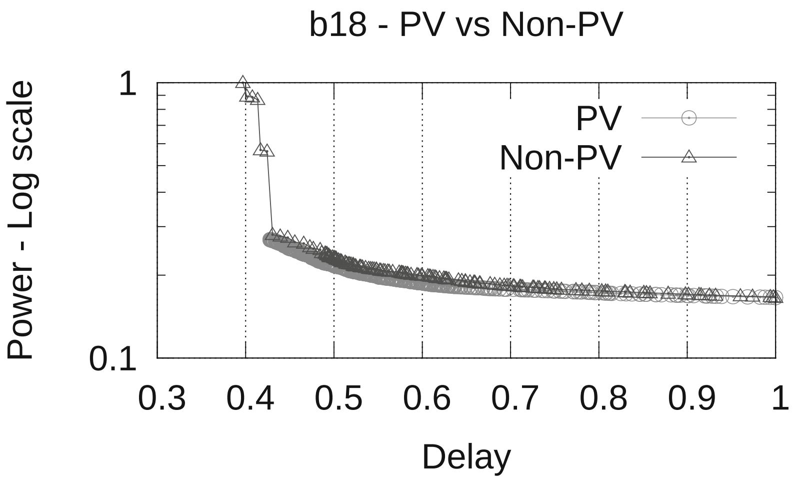


Fig. 10. Selected ITC-99 gate sizing result comparing the process variation-aware (PV) and non-process variation-aware (non-PV) techniques

The greatest savings were achieved from solutions where the non-PV sizing approach hit the timing wall earlier than the PV approach; however, this is not always the case as shown by one instance where the PV-aware sizing procedure achieved a lower performance delay target than its corresponding non-PV-aware approach as shown in Fig. 12(h), thus, high lighting the optimization uncertainty and room for potential improvement. It is also important to note that in circuit *c1355*, the target delays between 0.48 and 0.58 show solutions where the non-PV technique obtained superior results, which can be explained by speeding up a larger set of cells that belong in the ε-critical path early in the sizing effort, resulting with potentially more cells sized earlier than needed in order to achieve a lower performance target. However, sizing these ε-paths, enabled more efficient gate sizing, in terms of power-delay trade-offs, to be performed since the circuit was placed at a less-constrained sizing state, where each cell drives a smaller capacitive load, thus reducing the likelihood of getting stuck in a timing wall.

A timing wall scenario can be described where the cells that belong on the ε-critical paths are in such a state that increasing their size no longer improves circuit critical delay. Our experiments show that a gate sizing procedure that focus on the current critical path, result with over-optimizing a small set of critical paths, which in turn increases the convergence rate to an achievable target delay.

Tables 5 and 6 present yield results for selected ISCAS-85 and ITC-99 benchmarks. Yield results are acquired from computing the normalized delay and power values from 1000 PV-generated circuits for each benchmark. Timing and power values are reported for each benchmark from two obtained solutions in the previous gate sizing procedure. For fair comparison, each benchmark was optimized to meet an identical delay target. We report results obtained using two scenario optimization assumptions: (1) process variation-aware (PV), which uses the enhanced gate sizing algorithm presented in this Sect. 5.2; and (2) non-process variation-aware (nPV), which uses the standard gate-sizing algorithm presented in Sect. 5.1. The min, max, mean, standard deviation (std.), and variation values are presented for each benchmark. Variation is computed as $\frac{std.}{mean}$ and represents how much each yield result varies with respect to the mean. To summarize the results: ISCAS-85 benchmarks achieved a mean-power savings of 55 % (9 % min, 84 % max), whereas ITC-99 benchmarks achieved a mean-power savings of 25 % (2 % min, 74 % max).

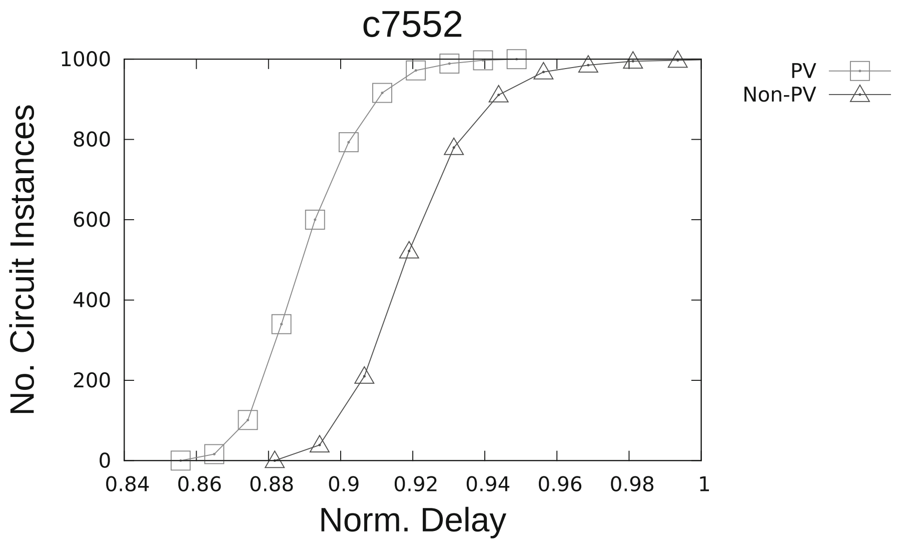
Figures 11 and 12 present the cumulative distribution graphs for their respective delay (left-graph) and power (right-graph) yield results for selected benchmarks, as presented in Tables 5 and 6. As shown, the circuits optimized using the PV-aware approach achieved significant power yield improvements with respect to target delay yields. Note that although the obtained solutions were originally optimized to satisfy the same target delay under both PV and non-PV gate sizing techniques, their representative yields demonstrate that the PV-aware approach achieved significant yield improvements in both delay and power. For instance, circuit *c432* achieved a normalized mean delay yield of 0.875 under the PV scenario compared to 0.920 for the non-PV scenario. Additionally, the delay variation factor of the PV scenario is 22 % less compared to the non-PV scenario. Collectively, the PV-aware solutions obtained lower variation factors in 14 out of the 19 studied benchmarks. Significant power reductions are also shown in the respective *c432* power yield graphs, achieving an 84 % mean-power savings (0.125 PV vs. 0.818 nPV), demonstrating the effectiveness of combining cell characterization and the gradual gate sizing technique over our presented gate sizing technique. Overall, the PV-aware solutions also achieved reduced variation factors in 14 out of the 19 benchmarks.

Table 5. ISCAS-85: yield results comparing 1000 generated process variation instances from a base circuit utilizing a: (1) process variation-aware (PV); or (2) non-process variation-aware (nPV) technique

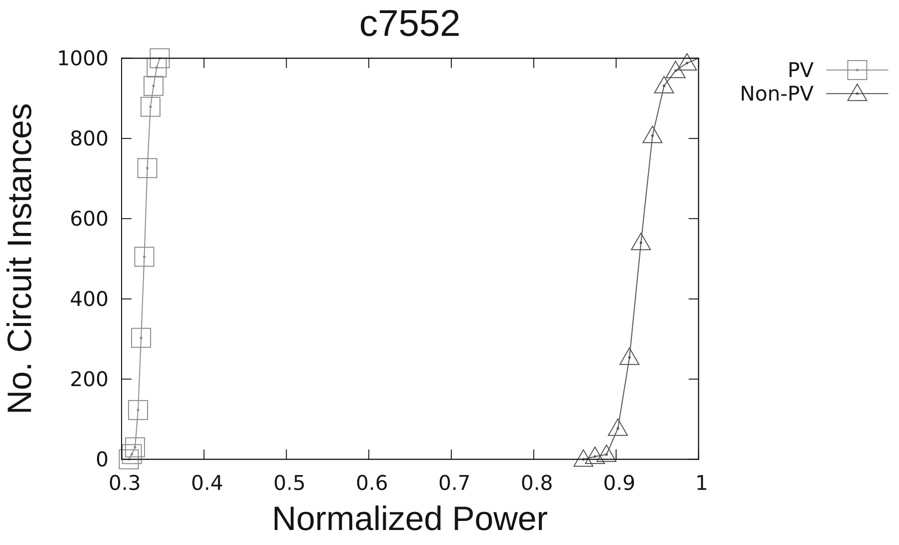
Benchmark	Scenario	Metric	Min	Max	Mean	Std	Variation
c1355	PV	Delay	0.853	0.983	0.900	0.022	0.956
	nPV		0.862	1.000	0.919	0.023	1.000
	PV	Power	0.738	0.946	0.829	0.029	1.000
	nPV		0.827	1.000	0.913	0.027	0.850
c1908	PV	Delay	0.869	0.980	0.914	0.017	0.943
	nPV		0.887	1.000	0.939	0.018	1.000
	PV	Power	0.368	0.457	0.406	0.016	0.980
	nPV		0.818	1.000	0.895	0.036	1.000
c2670	PV	Delay	0.870	0.994	0.915	0.018	1.000
	nPV		0.872	1.000	0.913	0.016	0.901
	PV	Power	0.195	0.233	0.211	0.007	0.830
	nPV		0.801	1.000	0.881	0.034	1.000
c3540	PV	Delay	0.877	0.966	0.921	0.014	0.974
	nPV		0.899	1.000	0.939	0.015	1.000
	PV	Power	0.385	0.504	0.441	0.019	1.000
	nPV		0.825	1.000	0.916	0.033	0.869
c432	PV	Delay	0.828	0.951	0.875	0.019	0.771
	nPV		0.858	1.000	0.920	0.026	1.000
	PV	Power	0.107	0.157	0.125	0.008	1.000
	nPV		0.706	1.000	0.818	0.053	0.992
c5315	PV	Delay	0.872	0.959	0.906	0.015	0.729
	nPV		0.882	1.000	0.928	0.021	1.000
	PV	Power	0.324	0.376	0.347	0.009	0.863
	nPV		0.861	1.000	0.930	0.028	1.000
c6288	PV	Delay	0.914	1.000	0.946	0.013	1.000
	nPV		0.890	0.968	0.926	0.012	0.940
	PV	Power	0.172	0.196	0.182	0.004	0.820
	nPV		0.860	1.000	0.916	0.024	1.000
c7552	PV	Delay	0.856	0.945	0.891	0.014	0.836
	nPV		0.882	1.000	0.920	0.017	1.000
	PV	Power	0.309	0.346	0.328	0.007	0.967
	nPV		0.860	1.000	0.929	0.020	1.000
c880	PV	Delay	0.822	0.954	0.872	0.021	0.875
	nPV		0.833	1.000	0.891	0.025	1.000
	PV	Power	0.640	0.828	0.724	0.037	0.828
	nPV		0.746	1.000	0.841	0.051	1.000

Table 6. ITC-99: yield results comparing 1000 generated process variation instances from a base circuit utilizing a: (1) process variation-aware (PV); or (2) non-process variation-aware (nPV) technique

Benchmark	Scenario	Metric	Min	Max	Mean	Std	Variation
b10	PV	Delay	0.720	0.900	0.789	0.029	0.885
	nPV		0.727	1.000	0.809	0.034	1.000
	PV	Power	0.486	0.866	0.608	0.057	0.961
	nPV		0.571	1.000	0.703	0.068	1.000
b11	PV	Delay	0.748	0.912	0.806	0.024	0.620
	nPV		0.759	1.000	0.849	0.041	1.000
	PV	Power	0.193	0.247	0.219	0.009	0.663
	nPV		0.682	1.000	0.841	0.052	1.000
b12	PV	Delay	0.749	0.946	0.815	0.035	1.000
	nPV		0.775	1.000	0.834	0.029	0.802
	PV	Power	0.285	0.389	0.332	0.019	0.837
	nPV		0.673	1.000	0.815	0.056	1.000
b13	PV	Delay	0.707	0.885	0.772	0.026	0.683
	nPV		0.760	1.000	0.846	0.042	1.000
	PV	Power	0.579	0.874	0.698	0.046	0.849
	nPV		0.630	1.000	0.756	0.058	1.000
b14	PV	Delay	0.816	0.925	0.847	0.016	0.974
	nPV		0.885	1.000	0.928	0.018	1.000
	PV	Power	0.546	0.664	0.587	0.017	1.000
	nPV		0.838	1.000	0.906	0.024	0.943
b15	PV	Delay	0.833	0.926	0.866	0.014	0.888
	nPV		0.893	1.000	0.938	0.017	1.000
	PV	Power	0.545	0.675	0.602	0.023	0.910
	nPV		0.789	1.000	0.867	0.037	1.000
b17	PV	Delay	0.896	0.984	0.929	0.013	1.000
	nPV		0.912	1.000	0.943	0.013	0.993
	PV	Power	0.864	0.978	0.903	0.016	0.995
	nPV		0.899	1.000	0.937	0.017	1.000
b18	PV	Delay	0.873	0.946	0.897	0.010	0.841
	nPV		0.919	1.000	0.949	0.013	1.000
	PV	Power	0.886	0.978	0.906	0.008	0.949
	nPV		0.907	1.000	0.930	0.009	1.000
b20	PV	Delay	0.814	0.901	0.845	0.015	0.665
	nPV		0.857	1.000	0.912	0.024	1.000
	PV	Power	0.863	0.967	0.902	0.015	0.875
	nPV		0.881	1.000	0.928	0.018	1.000

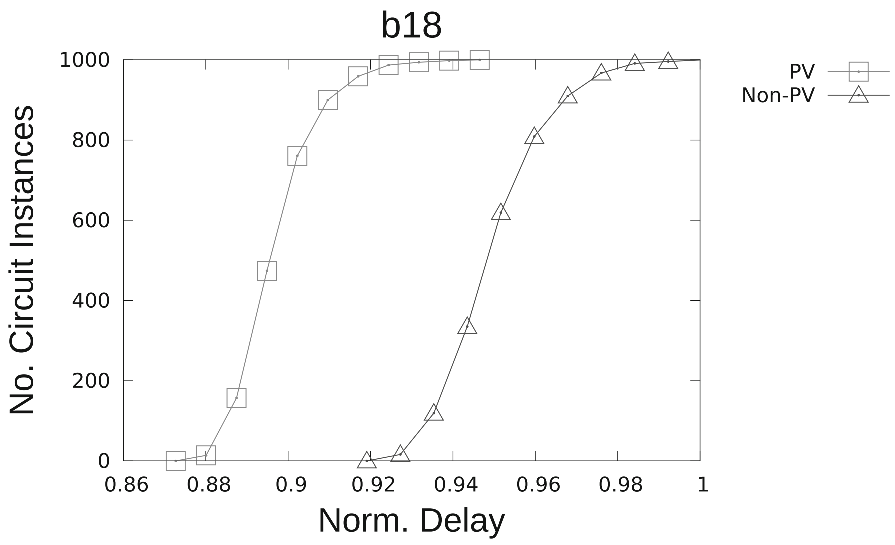


(a)

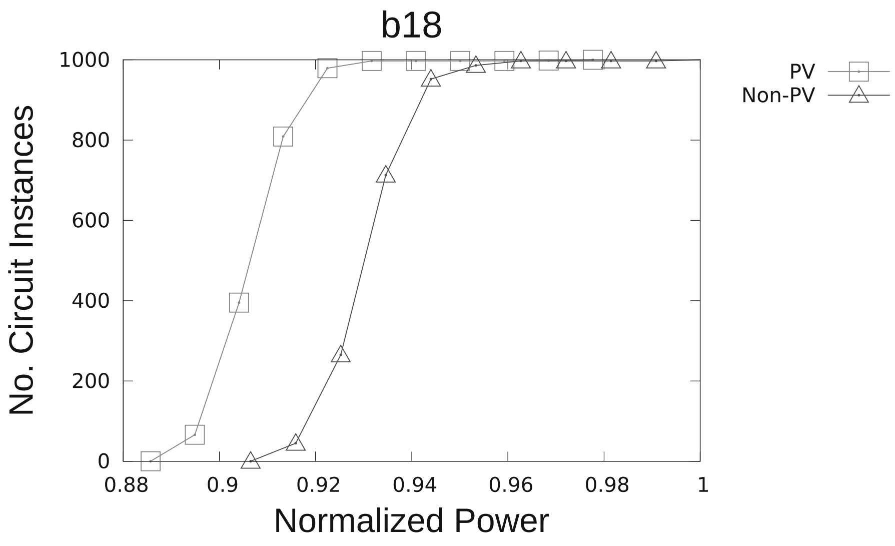


(b)

Fig. 11. Selected ISCAS-85 yield result comparing generated 1000 process variation instances with respect to identical target delay using generated solution from: (a) process variation-aware (PV); and (2) non-process variation-aware (non-PV)



(a)



(b)

Fig. 12. Selected ITC-99 yield result comparing generated 1000 process variation instances with respect to identical target delay using generated solution from: (a) process variation-aware (PV); and (2) non-process variation-aware (non-PV)

7 Conclusion

We have presented a new gate sizing approach that includes the switching activity (SA) and input vector control (IVC) to minimize overall energy. First, the new objective function has several ramifications on the optimization procedure, including the need for reiteration between gate sizing and input vector selection and freezing and unlocking of high-power gates. On a comprehensive set of benchmarks, from ISCAS-85/89, ITC-99, and arithmetic units, synthesized using 45 nm technology, we reduce average actual energy consumption by 30 %.

Next, we presented an extension of our gate sizing procedure for conducting delay and power yield optimization under uncertainty. Here, we further improve upon our presented gate sizing procedure to optimize a set of critically identified ε-critical paths in order to mitigate the impact of PV, thus, enabling more efficient delay and power trade-offs to be performed using our gradual gate sizing procedure. Under the yield optimization task, we compare generated solutions from selected circuits in ISCAS-85 and ITC-99 benchmark suites and show significant delay and power yield improvements of up to 55 % mean-power savings for ISCAS-85 circuits, and 25 % savings for ITC-99 circuits under equivalent timing targets.

We note that our presented techniques are generic in the sense that thermal impacts and multi-V_{th} can be easily addressed using the new optimization procedure.

References

1. Conos, N.A., Meguerdichian, S., Potkonjak, M.: Gate sizing in the presence of gate switching activity and input vector control. In: VLSI-SoC, pp. 138–143 (2013)
2. Shiyan, H., Ketkar, M., Hu, J.: Gate sizing for cell library-based designs. In: DAC, pp. 847–852 (2007)
3. Ozdal, M.M., Burns, S., Hu, J.: Gate sizing and device technology selection algorithms for high-performance industrial designs. In: ICCAD, pp. 724–731 (2011)
4. Joshi, S.: An efficient method for large-scale gate sizing. In: TCSI, pp. 2760–2773 (2008)
5. Ozdal, M.M., Amin, C., Ayupov, A., Burns, S., Wilke, G., Zhuo, C.: The ISPD-2012 discrete cell sizing contest and benchmark suite. In: ISPD, pp. 161–164 (2012)
6. Li, W.N.: Strongly NP-hard discrete gate-sizing problems. In: ICCD, pp. 1045–1051 (1993)
7. Agarwal, A., Chopra, K., Blaauw, D.: Statistical timing based optimization using gate sizing. In: DAC, pp. 400–405 (2005)
8. Nangate FreePDK 45 nm Library (2011). http://www.si2.org/
9. Coudert, O.: Gate sizing for constrained delay/power/area optimization. In: VLSI, pp. 465–472 (1997)
10. Srivastava, A., Sylvester, D., Blaauw, D.: Power minimization using simultaneous gate sizing, dual-Vdd and dual-Vth assignment. In: DAC, pp. 783–787 (2004)
11. Huang, Y.-H., Chen, P.-Y., Hwang, T.: Switching-activity driven gate sizing and Vth assignment for low power design. In: ASPDAC, pp. 24–27 (2006)
12. Abdollahi, A., Fallah, F., Pedram, M.: Leakage current reduction in CMOS VLSI circuits by input vector control. In: VLSI, pp. 140–154 (2004)
13. Yuan, L., Qu, G.: A combined gate replacement and input vector control approach for leakage current reduction. In: VLSI, pp. 173–182 (2006)

14. Lee, C., Potkonjak, M., Mangione-Smith, W.H.: MediaBench: a tool for evaluating and synthesizing multimedia and communicatons systems. In: MICRO, pp. 330–335 (1997)
15. Mudge T.: The SimpleScalar-Arm power modeling project. http://eecs.umich.edu/panalyzer/
16. Tsui, C., Au, R.Y., Choi, R.Y.: Minimizing the dynamic and subthreshold leakage power consumption using least leakage vector assisted technology mapping. In: VLSI Journal, pp. 76–86 (2008)
17. Li, L., Kang, P., Lu, Y., Zhou, H.: An efficient algorithm for library-based cell-type selection in high-performance low-power designs. In: ICCAD, pp. 226–232 (2012)
18. Hu, J., Kahng, A.B., Kang, S.H., Kim, M.-C., Markov, I.L.: Sensitivity-guided metaheuristics for accurate discrete gate sizing. In: ICCAD, pp. 233–239 (2012)
19. Asenov, A.: Random dopant induced threshold voltage lowering and fluctuations in sub-0.1 um MOSFET's: a 3-D atomistic simulation study. In: IEEE T-ED, pp. 2505–2513 (1998)