ECO Cost Measurement and Incremental Gate Sizing for Late Process Changes

JOHN LEE, UCLA
PUNEET GUPTA, UCLA

Changes in the manufacturing process parameters may create timing violations in a design, making it necessary to perform an Engineering Change Order (ECO) to correct these problems. We present a framework to perform incremental gate sizing for process changes late in the design cycle, and a method to create initial designs that are robust to late process changes. This includes a method to measure and estimate ECO cost, and to transform these costs into linear programming optimization problems. In the case of ECOs, on average, the method reduces ECO costs by an average of 89% in changed area compared to a leading commercial tool. Furthermore, the robust initial designs are, on average, 55% less likely to need redesign in the future.

Categories and Subject Descriptors: J.6 [**Computer Applications**]: Computer-Aided Engineering

General Terms: Design, Algorithms, Performance

Additional Key Words and Phrases: Gate sizing, incremental algorithms, ECO, linear programming

ACM Reference Format:
Lee, J., and Gupta, P. 2012. ECO Cost Measurement and Incremental Gate Sizing for Late Process Changes. ACM Trans. Embedd. Comput. Syst. 9, 4, Article 39 (July 2012), 11 pages.
DOI = 10.1145/0000000.0000000 http://doi.acm.org/10.1145/0000000.0000000

1. INTRODUCTION

With the aggressive production schedules in the semiconductor industry, the design of integrated circuits runs concurrently with the development of the manufacturing process itself. As a result, the exact manufacturing specifications change over the design period. Substantial changes in the specification may cause timing infeasibility issues, which require Engineering Change Orders, commonly referred to as ECOs, to fix. As a tool for ECOs, gate sizing is commonly used to incrementally update designs, as it is generally less intrusive than adjusting the placement or performing buffer insertion on the design, and can be more powerful than rerouting the design.

The nature of the ECO depends on when the updated information arrives in the product's development cycle. If the information arrives before substantial engineering time is spent, the product may simply be redesigned. In contrast, if significant time has been spent on the design, an ECO may be used that affects a minimal fraction of the design. When the violations are small, the design may be fixed manually; when the violations are large, they may be fixed using CAD tools in *incremental mode*, followed by manual tweaking to correct any remaining timing violations. The design is then verified using sign-off quality tools to verify the timing, power, crosstalk, and design rules, with more accuracy.

The change in the specifications can be substantial. For example, Figure 1 shows an example of process parameter change from April 2008 to March 2010, for a commercial 45nm process. The difference in these parameters is not negligible– the transistor off current (I_{off}) increases by

This work was supported in part by the NSF Award 811832 and by the SRC Task 1816.
Author's addresses: J. Lee and P. Gupta, Electrical Engineering Department, University of California at Los Angeles.

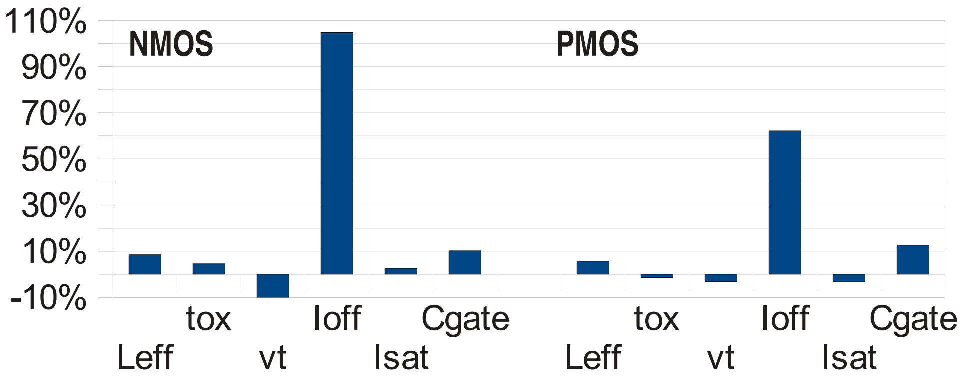

Fig. 1: Comparison of the 2008 and 2010 process specifications for a commercial 45nm process. The graph plots the percentage increase or decrease for several key parameters.

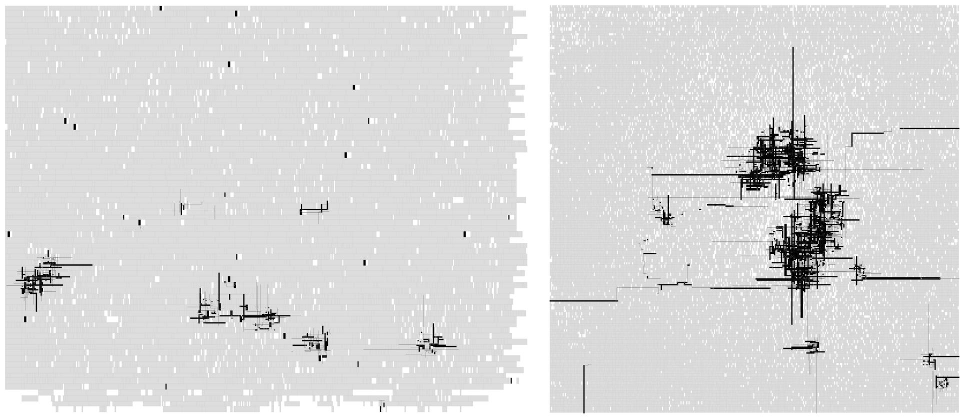

Fig. 2: Changed area caused by an ECO; $c_{\text{area}} = 27\mu\text{m}^2$, benchmark s38417 (left); and $c_{\text{area}} = 277\mu\text{m}^2$, benchmark mult (right).

over 80%, and the gate capacitance increases by approximately 10%. These two changes alone would have a large impact, by increasing the leakage power by over 80%, the dynamic power by approximately 10%, and the delay by approximately 10%. These are changes that may require substantial modifications in the design to correct the design according to its specifications.

In this paper, we focus on late-design cycle ECOs when the changes arrive after the design has been placed and routed, but before it is sent for fabrication. The changes in parameters may also result from retargeting a design to a different, but design-rule compatible, process[1]. We would like to (1) minimize the impact of the ECO, while maintaining a solution that is reasonably optimal after the process change is introduced, and (2) provide a method to modify designs to be robust against late process changes. In this paper, these goals are achieved by quantifying the ECO cost in terms of its area cost, and then approximating this relation as a function of layout parameters. The resulting model is fed into an optimization loop which minimizes the ECO cost and power while meeting the timing constraints. In comparison to the prior work in [Lee and Gupta 2010], an improved ECO

[1]Such multi-foundry sourcing is fairly common for large-volume designs.

area metric and a simplified version of the algorithm is presented in this paper, which has improved performance, and faster runtimes.

2. ECO COST

Research on ECO and incremental algorithms has focused on traditional costs such as wire-length, timing closure, and the number of changed nets (see for example [Chen et al. 2007; Dutt and Arslan 2006; Roy and Markov 2007]); however, they are too general to be used to distinguish between timing-feasible solutions with very similar power, but very different implementation cost.

In practice, the ECO cost is determined by the amount of time, in engineering work time and in tool hours, that is required to perform the ECO. This is the time is spent in checking and correcting: (1) timing errors, (2) problems with the layout, and (3) correcting design rule problems. Note that in modern designs and especially system-on-a-chip (SoC) designs, a large fraction of this verification may be manual.

As a measure of ECO that correlates to the costs in (1)-(3), we approximate these costs using an ECO area metric, c_{area}, that is the amount of layout area changed by the ECO. This area is computed over all layers of the design, and includes the amount of die area, in μm^2 that has been affected by:

— Cell resizing, movement or deletion
— Routing additions and deletions (interconnect and vias).

In this paper, these changes are measured using a commercial tool that compares the layout before and after the ECO change, and generates a list of gate changes and movements, and routing modifications. Next, a map of the changed die area is created, and the regions that are affected by the ECO (as in Figure 2) are marked. After all ECO changes are considered, the marked regions are added to produce the ECO area cost. The area ECO cost (c_{area}) is difficult to quantify without performing the ECO itself. These changes are the result of a chaotic interaction between the incremental design tool that is used and the current layout. However, there are intuitive rules that can be considered. The area cost is certainly related to the number of pins that are moved– each one of these pins require re-routing and reconnection. The difficulty in rerouting and reconnecting these pins is also related to the amount of free space in the routing layers above the cell. It is also important to consider the type of cell– some cells are tightly packed, which makes it difficult to access the pins. These ideas provide rules-of-thumb that designers can use to target low-ECO area designs.

For the purposes of guiding the optimization, we propose a method to estimate the effects of these rules-of-thumb on the ECO area cost as ($\hat{c}_{\text{area}}$) associated with changing a cell by performing a quick legalization-like placement check. This method first finds amount of free space around the current cell that is needed to accommodate the size change, and computes the required movements of the current cell and neighboring cells. This provides three pieces of information that are used to find the approximate ($\hat{c}_{\text{area}}$):[2]

— m_1: Number of *dislocated* pins
— m_2: Utilized area over pin bounding box (over all layers)
— m_3: The routing cost (from [Taghavi et al. 2010]).

The information m_1 and m_2 are related to the effects of this change on routing. The m_1 are the pins that are moved by the placement check, whose new and old locations *do not overlap*. This measure is important because the change in location will require a rerouting of the connections to the pins, and ECO area cost. The utilized area over the pin bounding box (m_2) is the area above the pin bounding box, the box containing all of the dislocated pins, that is used by the metal layers

[2]Other metrics such as congestion, net bounding boxes, number of changed cells, and the congestion on different metal layers, were also considered for estimating $\hat{c}_{\text{area}}$. The three measures used in this paper, m_1, m_2, and m_3 provided the best performance in terms of intuitive appeal, and accuracy.
Also note that the metrics used to estimate ECO cost (m_1 to m_3) differ from [Lee and Gupta 2010]. These improvements reduce the average normalized error by 4%.

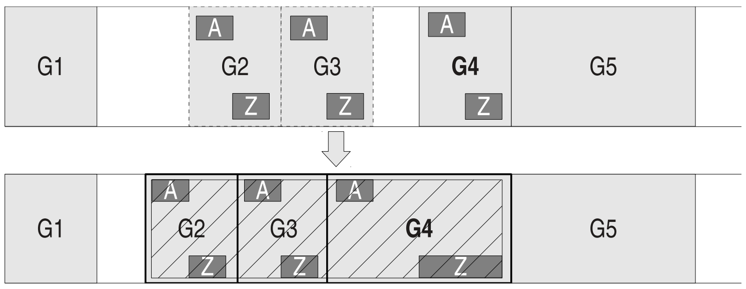

Fig. 3: ECO example to estimate c_{area}. Gate G4 changes from INV size 1 to INV size 2, dislocating cells G2 and G3. There are 6 pins that are moved by the change, but the number of dislocated pins, $m_1 = 5$, because pin G4/Z still overlaps with its old location.

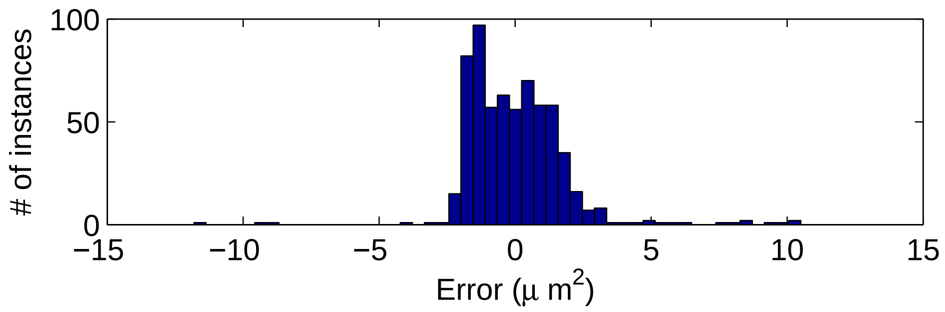

Fig. 4: Error histogram of the difference between the estimated ECO area values ($\hat{c}_{\text{area}}$) and the actual ECO area values (c_{area}) for 644 data points over the benchmark s35932.

for routing. Intuitively, larger values of m_2 indicate that it will be more difficult to reroute the m_1 dislocated pins, as the available space for routing is low, resulting in larger ECO costs.

The cost m_3 is a measure of the routability of a library cell called the *cell cost* [Taghavi et al. 2010], and is defined as:

$$[\text{Cell Cost}] = [\text{\# of pins}] + \sum_{\forall \text{pins} i} 2^{(2 - \frac{[\text{Area of pin i}]}{\Theta})} + \frac{1}{2} \sum_{\forall \text{pins} i} \sum_{\forall \text{pins} j \neq i} 2^{(2 - \frac{[\text{Area of the Bounding Box of pins i, j}]}{3\Theta})}. \tag{1}$$

In the above, Θ is the minimum cell pin width. The total cost m_3 is then the sum of the cell costs for all moved or re-sized cells. These parameters are then used in the linear model, $\hat{c}_{\text{area}}$, that estimates the true area cost as $\hat{c}_{\text{area}} = \sum_{i=1}^{3} a_i m_i + b$.

A sample of 644 ECO operations over the benchmark s35932 is used to fit the model, and a least-squares fit of the coefficients a_i is made. Each sample operation consists of changing the size of one gate, and recording the ECO cost, along with the values of m_i. The model parameters are:

a_1 = 0.183 $\mu\text{m}^2/\text{pin}$ a_2 = 4.721 a_3 = 0.123 b = 0.835 μm^2.

The quality of the fit is shown in Figure 4, which shows the errors between the estimate $\hat{c}_{\text{area}}$ and actual c_{area}. We shall see in Section 3.1 and in Tables II that the fidelity is high; minimizing the estimate $\hat{c}_{\text{area}}$ is effective in minimizing the actual ECO area cost.

We can use this information to estimate the cost of changing the size of a given cell. For example, consider the case in Figure 3. A quick placement check is done to find the values of m_1 to m_3. With the value $m_1 = 5$ (and assuming $m_2 = 0.25$ and $m_3 = 0.5$) the expression gives the estimate of $2.99\mu\mathrm{m}^2$.

These estimates are used to guide the ECO process. Gates in congested areas will result in large estimated ECO costs, as changing the gate will move many neighboring cells (resulting in large values for m_1 and m_3), and require re-routing in a congested area (m_2). Relying on changes with small ECO cost will help to make changes where free space is high and congestion is low.

3. SOLVING THE REDESIGN PROBLEM

Incorporating the ECO cost into the Linear Programming gate sizing framework in [Chinnery and Keutzer 2005] results in:

$$\begin{array}{ll} \text{minimize} & \sum_{i,k}(e_{ik} + \gamma p_{ik})y_{ik} \\ \text{subject to} & t_i + d_{i0} + \sum_k \delta_{ik} y_{ik} \leq t_j, \ \ \forall j \in \mathrm{fo}(i) \\ & t_i \leq T_{\max}, \ \ \forall i \in \mathrm{po} \\ & \sum_k y_{ik} \leq 1, \ \ \forall i \\ & 0 \leq y_{ik} \leq 1, \end{array} \tag{2}$$

which is applied iteratively. The variables are:

y_{ik}: Assignment variable of gate i to size k	
e_{ik}: ECO area cost estimate for y_{ik}	
t_i: Arrival time for gate i	d_{i0}: Current delay for gate i
δ_{ik}: Δ delay for y_{ik}	p_{ik}: Δ power for y_{ik}

We denote this algorithm LPECO-S, a *simplified* version of the LPECO from [Lee and Gupta 2010], that minimizes a weighted objective of power and ECO cost. The variables t_i, d_{i0} and δ_{ik} are related to the timing of the design, and they propagate the arrival times down the graph to enforce setup time constraints.[3] $\gamma = .05$, and is a factor used to consider the power, helping to break ties between gates with similar ECO costs. In contrast to [Chinnery and Keutzer 2005], to account for the downstream delays due to slew effects, the negative change in the slack is used as δ_{ik} in place of the actual delay change. Also, in contrast to [Lee and Gupta 2010], the restriction preventing neighboring gates to change is dropped.

The variable y_{ik} is an assignment variable that is 1 when gate i is size k in the solution, and 0 otherwise; the sum $\sum_k y_{ik}$ (for each i) is restricted to be less than or equal to 1 to prevent to assignment of a gate to multiple sizes. Note that for a given i, if all $y_{ik} = 0$, the current gate size is kept and not changed. The e_{ik} is the estimated ECO cost related to y_{ik}, if it were performed one gate at a time. The entire ECO cost is estimated by using the assumption that the ECO costs are additive.

As the number of gate sizing candidates is very large, we restrict the search to the gates that have negative slack, and the moves that improve slack (e.g. $\delta_{ik} < 0$). This means that the size of the problem is dominated by the number of possible moves, and not the size of the circuit. Furthermore, to consider the effect of fan-out load, gates are also considered if they are a fan-out of a critical gate. Fan-ins can also be considered to account for slew effects but we ignore them in our current experiments as they have little effect on delay for our benchmarks. Problem (2) may be infeasible when a large number of gate sizings is required to make the design timing-feasible. In these cases, the slack must be maximized iteratively, by solving (2) with $T_{\max}$ as the objective.

Also, when the solution to (2) has indeterminate assignments, e.g. the y_{ik} may be greater than 0, but less than 1, the gates are assigned using the same indeterminate assignment algorithm as in [Lee and Gupta 2010]. In this method, alternate cell options are considered that can provide the same slack improvement with less power and ECO cost.

[3]This formulation can also consider hold time constraints by adding a second set of timing variables, denoting the earliest arrival time for each gate. Also that design rules such as max transition and max capacitance can be handled in this formulation, by removing the assignments that violate these rules.

Table I: Benchmark Information for the nominal process

	70% Congestion				90% Congestion			
	cells	delay [ns]	power [μW]	die area [μm^2]	cells	delay [ns]	power [μW]	die area [μm^2]
c2670	912	0.589	8.0	1175	887	0.619	7.5	916
c3540	1538	1.118	10.1	1987	1423	1.053	12.6	1549
c5315	2038	1.046	14.5	2716	1899	0.973	16.4	2111
c6288	3451	2.290	23.7	3862	3128	2.226	22.8	2998
c7552	3029	0.925	25.7	3637	2773	0.957	23.1	2825
s13207	1183	0.612	22.8	3620	1083	0.618	22.6	2815
s35932	10570	3.054	144.5	23040	9842	4.899	136.9	17916
s38417	8820	1.793	133.0	21674	7744	1.740	129.7	16861
s38584	7908	4.366	103.7	16886	7131	2.946	98.8	13143
s5378	1286	0.923	14.2	2370	1052	0.881	13.5	1843
alu	13978	3.721	74.0	16242	12022	3.751	69.2	12640
mult	49141	6.095	558.2	54091	46701	7.324	401.3	42059

3.1. Experimental Results

This algorithm is tested on the ISCAS ‘85 and ‘89 benchmarks, a 64-bit multiplier, and the Open Cores ALU [OPE]. These benchmarks are synthesized to the Nangate 45nm Library [NAN], and placed, routed and optimized[4] on different sized dies to provide 70% and 90% congestion and experiment on the effects of congestion and free space on the ECO. Table I gives information about these benchmarks for the nominal process parameters.

The library is then adjusted for the following parameter changes, using the Liberty NCX tool[Synopsys 2010]

v_{t}: nmos -10%, pmos -5%	t_{ox}: nmos +5%, pmos -5%
c_{gate}: nmos +10%, pmos +10%	l_{eff}: nmos +5%, pmos +5%.

These changes are derived from a two year change in a commercial 45nm process as in [Lee and Gupta 2010], and they create a negative-slack timing violation that is repaired using the algorithm LPECO-S. For comparison, the algorithm is run without the ECO costs (LP No Eco Cost), and the commercial design tool is also used to repair the timing violation in the *post-route* incremental mode with the optimization effort set to high. The commercial tool has the ability to add buffers, on top of sizing gates, and while this provides an advantage over LPECO-S, we show that LPECO-S still performs better. All timing and power data in this paper is generated using this commercial design tool.

The algorithm LPECO-S is implemented using C++ and the linear programming solver in MOSEK [MOSEK ApS]. The ECO cost estimates are also programmed in C++, and the final ECO design is created using the commercial design tool.

Results are shown in Table II. The c_{area} and p_{l} represent the actual ECO area cost and leakage power, respectively. The “iters” column gives the number of iterations that the LPECO-S algorithm needs to find a timing-feasible solution. The slacks in the table are computed after the parameter changes. In all of the cases, the algorithm LPECO-S is able to find a timing feasible solution, while the commercial tool is unable to do so in 7 of the cases.

In the cases where both the LPECO-S and the commercial tool find a timing feasible solution, the LPECO-S provides significant reductions. On average, the area cost c_{area} improves by 93%; this performance is affected by the congestion; while the improvement is 99% for the 70% congestion benchmarks, it is 87% for the 90% congestion benchmarks.[5] This is due to the fact that it is more difficult to predict ECO area costs when the congestion is high, and the interactions between neighboring cells and interconnect increase. The difference in power between the commercial solution

[4]Note that this is a newer version of the tool used in [Lee and Gupta 2010]. In comparison to the benchmarks in [Lee and Gupta 2010], these benchmarks were more heavily optimized to produce a nominal design.

[5]Note that the difference in performance, compared to [Lee and Gupta 2010], is due to the improvements in the performance of the commercial tool.

Table II: Experimental Results comparing LPECO-S with the commercial tool

70% Congestion			LPECO-S				Commercial					LP (No ECO Cost)					
	$\text{slack}_{\text{init}}$ [ns]	p_{init} [μW]	slack [ns]	c_{area} [μm^2]	p_{l} [μW]	iter	slack [ns]	c_{area} [μm^2]	Δ	p_{l} [μW]	Δ	slack [ns]	c_{area} [μm^2]	Δ	p_{l} [μW]	Δ	iter
c2670	-0.028	8.0	0.000	0.028	7.97	2	0.000	1.51	98%	8.0	0.1%	0.001	12.55	100%	7.8	-2.2%	1
c3540	-0.053	10.1	0.002	5.249	10.26	3	-0.022	16.17	*	10.7	*	0.006	17.36	70%	9.96	-2.9%	3
c5315	-0.048	14.5	0.001	1.644	14.51	4	-0.022	7.17	*	14.5	*	0.001	7.84	79%	14.29	-1.5%	3
c6288	-0.113	23.7	0.000	4.596	23.87	2	-0.071	4.77	*	23.9	*	0.003	36.32	87%	22.69	-4.9%	3
c7552	-0.045	25.7	0.005	1.506	25.72	4	-0.002	16.53	*	25.9	*	0.002	43.41	97%	24.85	-3.4%	2
s13207	-0.020	22.8	0.095	0.014	22.84	1	0.095	1.65	99%	22.8	0.0%	0.095	0.01	0%	22.84	0.0%	1
s35932	-0.094	144.5	0.119	0.015	144.54	1	0.119	9.06	100%	144.6	0.0%	0.120	27.07	100%	144.31	-0.2%	1
s38417	-0.088	133.0	0.051	0.015	133.05	1	0.051	4.04	100%	133.1	0.0%	0.051	0.01	0%	133.05	0.0%	1
s38584	-0.084	103.7	0.344	0.029	103.69	1	0.344	19.93	100%	103.8	0.1%	0.004	31.42	100%	103.26	-0.4%	2
s5378	-0.038	14.2	0.050	0.013	14.21	1	0.050	1.15	99%	14.3	0.3%	0.050	0.01	0%	14.21	0.0%	1
alu	-0.139	73.9	0.015	0.013	73.95	1	0.015	6.13	100%	74.0	0.1%	0.015	0.01	0%	73.95	0.0%	1
mult	-0.316	558.2	0.154	0.013	558.20	1	0.154	14.82	100%	558.2	0.0%	0.149	37.85	100%	557.44	-0.1%	4
AVG	-0.089					1.8			99%		0.1%			61%		-1.3%	1.9
90% Congestion																	
c2670	-0.029	7.6	0.000	0.06	7.56	2	0.007	11.58	100%	7.7	1.4%	0.006	17.5	100%	7.25	-4.1%	6
c3540	-0.057	12.6	0.000	5.130	12.68	5	-0.016	31.07	*	13.1	*	0.002	47.19	89%	11.98	-5.5%	3
c5315	-0.047	16.4	0.001	0.070	16.44	2	-0.032	8.34	*	16.6	*	0.000	10.16	99%	16.21	-1.4%	1
c6288	-0.106	22.4	0.004	8.295	23.07	4	-0.086	3.15	*	22.9	*	0.005	47.28	82%	21.33	-7.6%	3
c7552	-0.040	23.2	0.013	2.636	23.15	2	0.010	9.82	73%	23.2	0.3%	0.003	10.83	76%	22.94	-0.9%	1
s13207	-0.018	22.6	0.023	0.013	22.62	1	0.088	1.10	99%	22.6	0.0%	0.023	0.01	0%	22.62	0.0%	1
s35932	-0.309	136.9	0.069	0.015	136.91	1	0.069	12.71	100%	136.9	0.0%	0.097	5.83	100%	136.84	-0.1%	2
s38417	-0.069	129.8	0.029	0.073	129.72	5	0.029	15.20	100%	129.8	0.1%	0.029	0.07	0%	129.72	0.0%	5
s38584	-0.128	98.8	0.495	0.014	98.80	1	0.778	9.78	100%	98.8	0.0%	0.579	12.99	100%	98.69	-0.1%	1
s5378	-0.025	13.6	0.048	0.079	13.54	1	0.049	3.52	98%	13.6	0.1%	0.031	0.01	-450%	13.54	0.0%	1
alu	-0.187	69.2	0.045	0.022	69.24	1	0.045	4.32	99%	69.3	0.1%	0.045	0.46	95%	69.24	0.0%	2
mult	-0.028	401.3	0.350	372.359	401.31	1	0.348	427.3	13%	401.4	0.0%	0.035	277.1	-34%	401.22	0.0%	1
AVG	-0.087					2.2			87%		0.2%			21%		-1.6%	2.3

*denotes infeasible designs

and the LPECO-S solution is very small (.17%), indicating that the ECO cost is needed to distinguish between solutions that are similar in power, but have different ECO implementation costs.

In the comparison with the ECO cost disabled (LP Without ECO Cost), LPECO-S yields a significantly better ECO area cost in the majority of cases. In the 70% and 90% congestion cases, the area cost reduction was, on average, 61% and 21% respectively. There are a couple cases in the 90% where the LP Without ECO Cost performs better than the LPECO-S; however, these are not shortcomings of the algorithm, and have more to do with the difficultly in predicting the ECO cost at high congestion. In the s5378 case, the absolute difference is negligible (.06μm^2), and in the mult case, the algorithm is unable to predict the effects of incremental routing; the LPECO-S changes just one gate, from size 1 to size 2, while the LP Without ECO sizes 9 gates over an area with similar routing utilization (m_2). This difference is primarily due to routing changes, and is a comment on the difficulty of predicting routing changes.

The LP Without ECO Cost is able to improve the power of the design by an average of 1.5%. This is because the objective here is to fix the timing violation with the greatest power benefit. However, this is not ideal for the ECO case, as the focus is on minimal disturbance, and the greatest power savings may result in larger ECO costs (e.g. c6288 90% congestion). Furthermore, the power difference is negligible in the larger designs.

The runtime for this algorithm is dominated by the interface from the commercial tool to LPECO, which is needed to transfer timing information and gate sensitivity information. This sensitivity information is needed for any sizer, as the comparisons between competing gates must be made in the process of optimization. Each iteration of LPECO-S takes between 6 and 280 seconds, while solving the linear program in LPECO-S takes between .02 to 2.1 seconds for all benchmarks (excluding the time used by the commercial physical design tool). This is significantly faster than in [Lee and Gupta 2010], which required up to 103 seconds. In comparison, running the LP (without the ECO cost) takes between 1 and 71 seconds per iteration. The runtime of the commercial tool is comparable to the runtime needed to by the same commercial tool to perform the ECO, and ranges between 24 seconds and 23 minutes.

4. CREATING INITIAL DESIGNS

In some cases, there may be several target foundries that may be targeted for production, or there may be uncertainty in the manufacturing process parameters; there may be an idea of which parameters may fluctuate, and which parameters would be controlled well in future. These situations motivate the initial design problem, where an initial design is created that can tolerate future manufacturing process fluctuations.

We consider the following formulation of this problem. Suppose, as a starting point, we have an original, optimized design that has undergone placement and routing, and is timing-feasible in the nominal case. The information on potential manufacturing-process changes in the form of corners, scenarios, or samples. As designing for all possible cases results in an overly conservative design with a large power, the goal of the initial design is: (1) the resulting design is timing feasible in the nominal corner; (2) the difference between the power of the original design, ($p_{y_{\text{orig}}}$), and the power of the new initial design is within a tolerance β; and (3) the need for a future ECO is reduced.

As a heuristic to meet these goals, we propose the following linear programming problem to solve the initial design problem:

$$\begin{array}{ll} \text{minimize} & t_{\max} \\ \text{subject to} & \sum_{i,k} p_{ik} y_{ik} \le (\beta \cdot p_{y_{\text{orig}}}) \\ & t_i + \\ & \frac{1}{N+1} \sum_{n=0}^{N} (d_{i0}^{(n)} + \sum_k \delta_{ik}^{(n)} y_{ik}) \le t_j,\ \forall i \in \text{fo}(j) \\ & t_i \le t_{\max},\ \ \forall i \in \text{po} \\ & \sum_k y_{ik} \le 1,\ \forall i,\ \ 0 \le y_{ik} \le 1. \end{array} \tag{3}$$

The meanings of the variables y, δ and t are the same as in the LPECO-S formulation in (2). N is the total number of corners that are used, and n is the index used for the corners, with $n = 0$ denoting the nominal corner. The superscript $^{(n)}$ refers to the corner associated with the delay $d_{i0}^{(n)}$ or change in delay $\delta_{ik}^{(n)}$. Cell options that are delay improving in the nominal process ($\delta_{ik}^{(0)} < 0$) are considered as candidate cell changes. This formulation is similar in concept to [Boyd et al. 2005], where the delay for each gate is converted to a statistical delay. In this case, the manufacturing uncertainty is accounted for by adjusting the delay to be the average over the given scenarios. In contrast to (2) where the power plays a role in the objective, in this formulation it is used as a constraint.

Note that the input to the algorithm is a design that is timing-feasible in the nominal scenario. Also, as in Section 3, only moves that are delay-improving in the nominal process ($\delta_{ik}^{(0)} < 0$), are considered.

In the above, the variation information is assumed to be in the form of N corners, scenarios or samples. This flexible way to describe variations is useful when the information on the manufacturing parameters is scarce; there may not be accurate distributions available for modeling future variations. These corner-type specifications can then describe the kinds of variations that the designer would like to hedge against.

The algorithm (3) is similar to a statistical version of guardband. Given an amount of power β that the designer is willing to spend, the design maximizes the average slack over all of the scenarios using gate sizing. In effect, the *expected slack* is maximized to decrease the need for future ECO.[6] This algorithm is not applied iteratively and is run only once.

After (3) is solved, the y_{ik} are mapped to gate sizes by applying the methods in [Lee and Gupta 2010]. The indeterminate assignments are remapped if possible, and the candidates are sorted by sensitivity with values of $y_{ik} > 0.01$ as eligible for change. The changes are made until the power budget (e.g. the tolerance β) is met. Furthermore, each gate sizing is checked to ensure that it does not cause timing violations in the nominal process parameters, and is skipped if timing violations are created.

Note that this work is different from work in statistical gate sizing. In this situation, the manufacturing process changes may be impossible to predict using distributions, and the power and timing effects may be impossible to model statistically. This method provides a method to create initial designs with little statistical information, that are robust to manufacturing process changes, and is also simple enough to implement on top of current tools.

4.1. Experimental Results

This algorithm is tested on the benchmarks in Table I. The manufacturing process changes are assumed to be random variables with zero-mean Gaussian distributions, and the following standard deviations:

v_{th}: 5% t_{ox}: 2.5% C_{gate}: 2.5% l_{gate}: 2.5%.

The variations are the same across all gates (e.g. all transistors have the same increase in v_{th}, t_{ox}, C_{gate} and l_{gate}). However, the variations between the PMOS and NMOS transistors for the v_{th} and t_{ox} parameters are considered to be independent. This model may be pessimistic, as more information may be available, such as the direction of the variation. For example, the foundry might give the current and target PMOS v_{th}, implying that the final value would be between the current and the target values.

10 samples (set 1) are randomly generated according to the distribution above and are used in the LPECO-ID algorithm. These samples, along with the nominal process parameters, are used to create the initial design. A *separate* set of 10 different independent samples (set 2) is generated using the same distribution to evaluate the quality of the LPECO-ID algorithm. The two sets of samples are generated independently to simulate a realistic design condition. While a rough idea

[6]While ECO area costs can be added to this formulation, we find that the improvements are not significant, as improving the slack and the future feasibility is the dominating effect.

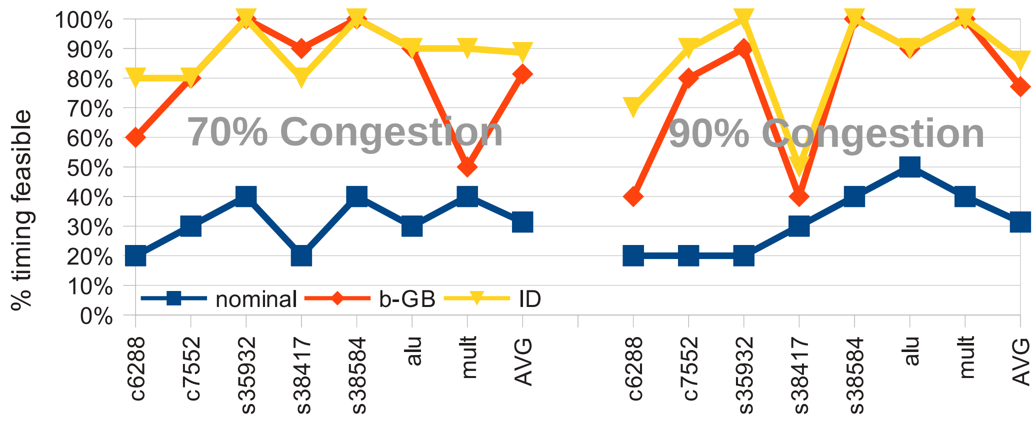

Fig. 5: Results comparing the feasibility of the original design, a modified design using a β-guardband, and the Initial Design Method (ID). The Initial design method improves the feasibility substantially.

of the variations for the manufacturing process parameters may be known, the actual values are unavailable until after the initial design is set.

This initial design method (LPECO-ID) is implemented using C++ and the linear programming solver in MOSEK [MOSEK ApS]. The ECO cost estimates are also programmed in C++, and the final ECO design is created using the commercial design tool for each of the manufacturing process variations. As a comparison, the same β budget is used to create a guardband β-GB by maximizing the slack in the nominal scenario.

The results in Figure 5 show that the LPECO-ID method drastically reduces the need to perform an ECO compared to the commercial tool. An ECO is needed just 13% of the time ID algorithm, while it is needed 21% of the time with β-GB, and 68% with the original design. In the s38417 70% congestion case, the β-GB performs slightly better, but this the only exception. This shows that this method is effective in hedging against future changes.

5. CONCLUSION

In this paper, we present the idea of ECO cost to quantify the amount of time that is needed to validate an ECO operation. We then propose a novel method for performing ECO gate sizing, and give models for the ECO that can be incorporated into the optimization procedure. This leads to results that outperform a leading commercial design tool in reducing the amount of area that is changed by the ECO by an average of 89%. In addition, a novel method for creating initial designs is presented that drastically reduces the probability that a redesign is needed in the future, between 10% and 80%.

REFERENCES

Available from http://www.opencores.org.

Nangate Open Cell Library v1.3. Available from http://www.si2.org/openeda.si2.org/projects/nangatelib.

BOYD, S., KIM, S., PATIL, D., AND HOROWITZ, M. 2005. Digital circuit optimization via geometric programming. *Operations Research 53,* 6, 899.

CHEN, Y.-P., FANG, J.-W., AND CHANG, Y.-W. 2007. Eco timing optimization using spare cells. In *Proc. Int. Conf. Computer-Aided Design*. 530–535.

CHINNERY, D. G. AND KEUTZER, K. 2005. Linear programming for sizing, vth and vdd assignment. In *Proc. Int. Conf. Low Power Electronics and Design*. 149–154.

DUTT, S. AND ARSLAN, H. 2006. Efficient timing-driven incremental routing for vlsi circuits using dfs and localized slack-satisfaction computations. In *Proc. Design, Automation and Test in Europe*. 768–773.

LEE, J. AND GUPTA, P. 2010. Incremental gate sizing for late process changes. In *Proc. Int. Conf. Computer Design*. 215 –221.

MOSEK APS. The MOSEK Optimization Tools Version 5.0. Available from http://www.mosek.com.

ROY, J. AND MARKOV, I. 2007. ECO-system: Embracing the Change in Placement. *IEEE Trans. on Computer-Aided Design 26*, 12, 2173–2185.

SYNOPSYS. 2010. Liberty ncx d-2009.12-sp3. http://www.synopsys.com/.

TAGHAVI, T., LI, Z., ALPERT, C., NAM, G., HUBER, A., AND RAMJI, S. 2010. New placement prediction and mitigation techniques for local routing congestion. In *Proc. Int. Conf. Computer-Aided Design*. 621–624.

Received December 2011; revised April 2012; accepted July 2012