Incremental Gate Sizing for Late Process Changes

John Lee
Electrical Engineering
Department
UCLA
lee@ee.ucla.edu

Puneet Gupta
Electrical Engineering
Department
UCLA
puneet@ee.ucla.edu

ABSTRACT

We present a new framework to measure ECO cost resulting from process changes late in the design cycle and perform incremental gate sizing to minimize this layout and timing verification cost. Compared to a commercial solver, ECO costs are reduced up to 99% in changed area, and up to 96% in non-critical changed pins.

1. INTRODUCTION

The design of integrated circuits runs concurrently with the development of the manufacturing process itself, and as the manufacturing process will change, Engineering Change Orders will be required to modify the design.

If the ECO information arrives before substantial engineering time is spent, the product may simply be redesigned. Later arrivals would be wise to employ an ECO that affects a minimum fraction of the design and very late arrivals require the use of back-end methods, such as utilizing spare cells, and re-routing the interconnect of a design.

Research on incremental algorithms has been ongoing for nearly two decades ([4],[5], [6]). However, as far as the authors know, the subject of ECO gate sizing and minimizing the impact of an ECO has received no attention.

This paper studies late-design cycle ECOs, which occur before fabrication. The cost impact of the ECO should be minimized while maintaining a solution that is reasonably optimal. The contributions of this paper are: (1) measures to quantify ECO cost in terms of timing and area change; (2) a new algorithm to perform discrete sizing with ECO cost estimates using linear programming; and (3) comparisons with a commercial physical design tool that illustrates the superiority of the proposed approach.

1.1 How specifications can change

Specifications can change substantially. For example, Figure 1 shows the percentage 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 over 80%, and the gate capacitance increases by approximately 10%. These two changes can increase the leakage power by over 80%, the dynamic power by approximately 10%, and the delay by approximately 10%. With this uncertainty in manufacturing specifications, it becomes important to research algorithms that adjust designs to account for these changes.

2. ECO COST

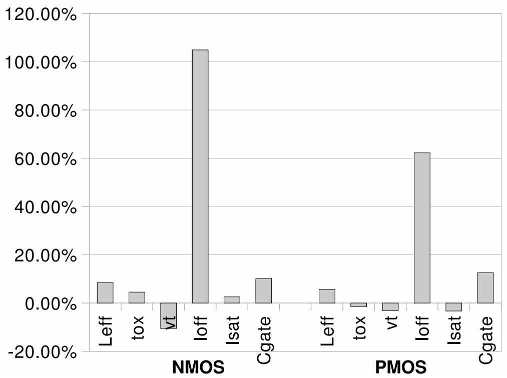

Figure 1: Comparison of the 2008 and 2010 process specifications for a commercial 45nm process. The graph plots the percentage increase or decrease for several key parameters.

Research on ECO and incremental algorithms has focused on traditional costs – wire-length, timing closure, and the number of changed nets. These metrics do not measure the cost needed to implement the design.

This paper focuses on the gate sizing problem as it is one of the most flexible and widely used methods available. It is less intrusive than adjusting the placement of the design, and more powerful than rerouting the design.

The ECO cost is related to the time that is required to perform the ECO:

1. Checking and correcting the timing: how much of the design must be rechecked for timing validity, and how much time is need to fix any detected errors? Note that in modern system-on-a-chip (SoC) designs, a large fraction of this verification may be manual.
2. Checking and correcting the layout: is the resulting layout manufacturable with high yield?
3. Checking and correcting design rules: are there violations in the electrical or layout characteristics (maximum capacitance, slew, wire density)

Thus, it is important to find a measure of ECO, ECO($\cdot$), that correlates to the costs in (1)-(3). We approximate the costs using two measures:

1. c_{area}: The area change from the ECO: the amount of layout area that changes. This includes area that is changed by cell changes, cell movement, and routing

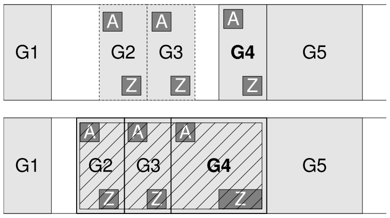

Figure 2: Gate G4 changes from INV size 1 to INV size 2, dislocating cells G2 and G3. All the pins are affected by the change ($m_1 = 6$), but $m_2 = 5$ because pin G4/Z overlaps with its old location. The cross-hatched area is the pin bounding box area m_3.

changes. This area is computed over all layers of the design.

2. c_{timing}: The number of non-critical pins that are in the fan-in or fan-out cones of the ECO-changed cells. This is used to measure the ECO cost related to unintended timing changes.

The ECO cost is a function of the circuit layout, the interconnect routing, and the type of change that is needed. The timing ECO cost (c_{timing}) can be predicted by counting the number of non-critical pins in the fan-out and fan-in cones of the changes. In contrast, the area ECO cost is difficult to quantify without performing the ECO itself. This cost is the result of a chaotic interaction between the incremental design tool and the current layout.

For the purposes of guiding the optimization we construct and estimate ECO area cost ($\hat{c}_{\text{area}}$) from the following information about a potential change:

- m_1: Number of affected pins
- m_2: Number of dislocated pins (old locations and new locations do not overlap)
- m_3: Pin bounding box area
- m_4: Utilized area over pin bounding box (routing over all layers)

This information is obtained by using a quick placement check that finds the amount of cells that must be moved to find free space for the potential ECO. These parameters are used in a linear model as $\hat{c}_{\text{area}} = \sum_{i=1}^{4} a_i m_i + b$.

A sample of ECO operations is made to fit the model, and a least-squares fit of the coefficients a_i is made. The values are $a_1 = 0.0367\ \mu\text{m}^2/\text{pin}$, $a_2 = 0.186\ \mu\text{m}^2/\text{pin}$, $a_3 = 5.35$, $a_4 = 9.65$, and $b = .264\ \mu\text{m}^2$. The quality of the fit is shown in Figure 3, which shows that the fit has a substantial error, but the fit clearly identifies an increasing trend in the data.

These estimates can be use to avoid changes in congested areas. When there are many neighboring gates closeby, m_1 to m_3 (and hence estimated ECO cost) will be large. Areas with high routing congestion will have a large m_4 resulting in a large estimated ECO cost, and avoidance by the algorithm.

3. SOLVING THE REDESIGN PROBLEM

Suppose we would like to solve the incremental problem: *given the current set of sizes x, find a suitable adjustment y which is the solution to:*

$$\begin{array}{ll} \text{minimize} & \text{Power}(y) + \gamma\text{ECO}(y; x) \\ \text{subject to} & \text{Delay}(y) \leq T_{\max} \end{array} \qquad (1)$$

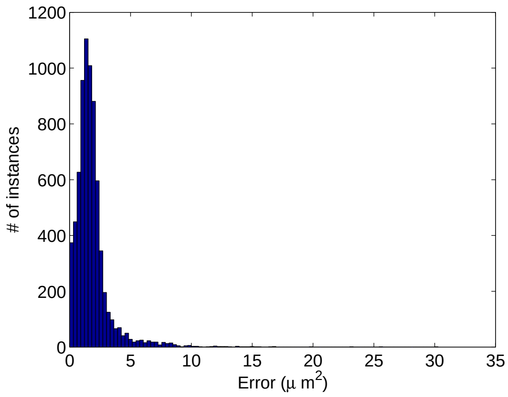

Figure 3: Error histogram of the difference between the estimated ECO area values ($\hat{c}_{\text{area}}$) and the actual ECO area values (c_{area}) for 7274 data points over the ISCAS '85 benchmarks.

This is the fundamental ECO problem – how can the power and ECO costs be juggled to meet the timing constraint? The term $\text{ECO}(y; x)$ measures the amount of change or the difference in the designs x and y, in terms of an ECO cost. As γ becomes large, this cost becomes more significant.

We solve this problem, with the following assumptions:

1. The ECO costs are additive (the total ECO is the sum of the costs of the individual ECOs)
2. Out of every two connected gates, at most one gate should change its size (see Section 3.2)

This results in the following linear programming approximation to (1):

$$\begin{array}{ll} \text{minimize} & \sum_{i,k} p_{ik} y_{ik} + \gamma\text{ECO}(y; x) \\ \text{subject to} & t_i + d_{i0} + \sum_k \delta_{ik} y_{ik} \leq t_j, \quad \forall i \in \text{fo}(j) \\ & t_i \leq T_{\max}\,, \quad \forall i \in \text{po} \\ & \sum_k y_{ik} \leq 1, \quad \forall i \\ & \sum_k y_{ik} + \\ & \ldots\ \sum_{j \in \text{fo}(i)} \sum_k y_{jk} + \sum_{j \in \text{fi}(i)} \sum_k y_{jk} \leq 1, \quad \forall i \\ & 0 \leq y_{ik} \leq 1 \end{array} \qquad (2)$$

The variables are:

- t_i: Arrival time for gate i
- d_{i0}: Current delay for gate i
- δ_{ik}: Change in the delay of gate i under size k
- y_{ik}: Assignment variable of gate i to size k
- p_{ik}: Power cost of changing gate i to size k

We call this algorithm LPECO. This algorithm finds an assignment of sizes to gates that minimizes a weighted objective of power and ECO cost.

As the number of possible moves is very large, we restrict the search to the nodes that have negative slack, and the moves that improve slack (e.g. $\delta_{ik} < 0$). Furthermore, to consider the effect of fan-out load, nodes are also considered if they are a fan-out of a critical node. Fan-ins are not considered as they have minimal effect on the delay.

The constraint $\sum_k y_{ik} \leq 1$ prevents the assignments of gate i from add up to more than 1. Due to the properties of linear programming, this constraint will also ensure each gate i will have at most one k with $y_{ik} > 0$, with the remainder of the k having $y_{ik} = 0$.[1] However, the value of the assignment may not be 1.

The constraint

$$\sum_k y_{ik} + \sum_{j \in \text{fo}(i)} \sum_k y_{jk} + \sum_{j \in \text{fi}(i)} \sum_k y_{jk} \leq 1 \tag{3}$$

is used to help enforce assumption 1, that only one gate out of every two connected gates will change size. However, this does not guarantee that only one gate out of every neighboring pair will be assigned, and we will consider these indeterminate cases in Section 3.4.

If we assume that the delay models are sufficiently accurate (to some minimum tolerance), the solution will give a lower bound on the optimal assignment:

$$\sum_{i,k} p_{ik} y^{\star}_{ik} + \gamma \text{ECO}(y^{\star}; x) \tag{4}$$

3.1 Incorporating ECO costs

Introducing c_{area} into the optimization problem 2 is straightforward, as the model in Section 2 can be used to estimate the area cost of moving gate i to size k (e_{ik}). This gives:

$$c_{\text{area}} = \sum_{\forall i,k} e_{ik} y_{ik} \tag{5}$$

which can be added to the objective of (2).

Incorporating the timing cost can be incorporated as a series of additional constraints and an extra term in the objective. The additional constraints are:

$$\begin{array}{ll} \tau_i^{\text{fo}} \leq \tau_j^{\text{fo}}, \ \ \forall i \in \text{fo}(j) & \tau_k^{\text{fi}} \leq \tau_j^{\text{fi}}, \ \ \forall j \in \text{fi}(k) \\ \tau_i^{\text{fi}} \leq \tau_i, \ \ \forall i & \tau_i^{\text{fo}} \leq \tau_i, \ \ \forall i \\ \sum_{\forall k} y_{ik} \leq \tau_i^{\text{fi}}, \ \ \forall i & \sum_{\forall k} y_{ik} \leq \tau_i^{\text{fo}}, \ \ \forall i. \end{array} \tag{6}$$

Variables τ_i^{fo}, τ_i^{fi} and its related constraints ensure that the timing fan-out and fan-in cones are marked to be included in the timing cost. The variable t_i along with its related constraints ensure that the timing cost will be included if it is in the fan-in cone or fan-out cone of an ECO node. The constraints involving y_{ik} ensure that for any ECO node, the fan-out and fan-in cones are counted. c_{timing} is added to the objective as:

$$c_{\text{timing}} = \sum_i r_i \tau_i \tag{7}$$

where r_i is the number of non-critical pins on node i.

3.2 Restrictions on neighboring nodes

The assumption that "out of every two connected gates, at most one gate should change its size" is made because we assume that the ECO sizing changes are small changes over the entire circuit. Thus, because number of changes are small, we may assume that connected gates are not likely to change.

3.3 Slack Maximization

Problem (2) is infeasible when the amount of negative slack is too large to be fixed in one iteration. In these cases, the slack must be maximized iteratively, until problem (2) becomes feasible. This is done by changing the objective in (2) to minimizing $T_{\max}$. In this paper, we iterate until a timing feasible solution is found, with a maximum number of iterations set at 10.

3.4 Indeterminate assignments

The solution to (2) may have indeterminate assignments, e.g. the y_{ik} may be greater than 0, but less than 1. In these cases, a decision must be made as to whether a gate should be changed, and if so, which size it should be assigned to.

A guideline for the indeterminate assignments in the problem (2) can be derived from the lower bound equation (4). As this equation is linear, we can approximate the suboptimality as:

$$\sum_{\forall y_{ik} > 0} (p_{ik} + \gamma_{\text{area}} e_{ik})(1 - y^{\star}_{ik}) + c_{\text{timing}}. \tag{8}$$

This suboptimality comes from the difference between the continuous and the integer solutions to the problem.

We can reduce this gap by considering other sizes that may reduce the suboptimality in (8). Although the term c_{timing} is the same for any size assignment to gate i, the left side of the expression can be reduced by considering alternate assignments. Formally, if we are given an indeterminate assignment y_{ik}, the suboptimality is minimized over k by choosing the size s as:

$$s = \operatorname*{argmin}_{\{j| \ \delta_{ij} \geq y_{ik} \delta_{ik}\}} \ p_{ij} + \gamma_{\text{area}} e_{ij} \tag{9}$$

In the case of the slack maximization problem, we can also use (9), although there is no lower bound analysis available in this case. However, this can help the slack minimization algorithm choose solutions that have smaller power and ECO costs.

In a small minority of cases, the algorithm will assign neighboring gates. This is fixed with a greedy algorithm that creates the assignments in order of increasing sensitivity (Δobjective/Δslack). If a gate has a neighbor that has already been mapped, the gate is skipped and left unmapped.

4. EXPERIMENTAL RESULTS

This algorithm is tested on the ISCAS '85 benchmarks and the Open Cores ALU[1], which are synthesized to the Nangate 45nm Library[2], and optimized using a leading commercial design tool. The library is then adjusted for the following parameter changes, using a different commercial tool as 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 2 year change in a commercial 45nm process, and create a negative slack, or timing violation, that is repaired using the algorithm LPECO in Section 3 and the commercial design tool in *post-route* mode and the optimization effort set to high. All timing data in this paper is generated using this commercial design tool.

The algorithm LPECO is implemented using C++ and the open source linear programming solver lp_solve [3]. The final ECO design is created using the commercial design tool.

Results are shown in Table 1 for the congestion targets 70% and 90%. The c_{area}, c_{timing} and p_{l} represent the actual ECO area cost, ECO timing cost and leakage power, respectively. When a timing feasible design cannot be found, the

[1] A solution with multiple $y_{ik} > 0$ for a given k can be improved by consolidating the y_{ik} into the choice with the better objective vs. slack trade-off.

Table 1: Experimental Results

70% Congestion										
			Commercial				LPECO			
	slack$_{\text{initial}}$ (ns)	p_{initial} (μW)	slack (ns)	c_{timing} (pins)	c_{area} (μm^2)	p_{l} (μW)	slack (ns)	c_{timing} (pins)	c_{area} (μm^2)	p_{l} (μW)
c1355	(.022)	12.21	(.026)	78	70.19	13.11	(.016)	78	51.97	12.68
c1908	(.031)	12.88	(.045)	286	16.18	12.98	(.029)	113	.1	12.88
c2670	(.011)	17.69	(.015)	550	71.21	18.24	.000	479	53.21	17.84
c3540	(.049)	26.51	(.070)	657	76.65	26.72	(.056)	660	27.24	26.92
c5315	(.043)	29.24	(.055)	659	20.38	29.38	(.049)	423	6.03	29.41
c6288	(.083)	53.28	(.099)	425	79.81	53.63	(.092)	404	55.08	54.11
c7552	(.036)	45.09	(.037)	1139	76.33	45.44	(.034)	1388	40.47	45.63
alu	(.123)	168.46	(.045)	8733	279.8	168.63	(.097)	7861	65.36	168.92

90% Congestion										
c1355	(.018)	9.29	(.009)	330	63.3	10.65	.002	327	34.3	9.43
c1908	(.036)	8.73	.001	417	43.0	9.50	.008	329	27.5	8.78
c2670	(.015)	13.27	.000	510	34.5	13.73	.002	167	17.3	13.26
c3540	(.038)	19.23	.005	931	106.31	20.52	.025	790	43.5	19.45
c5315	(.048)	26.56	(.004)	1030	83.12	27.16	.005	964	48.2	26.64
c6288	(.085)	38.87	(.004)	1413	149.48	41.00	.002	1002	102.5	38.69
c7552	(.048)	34.82	.003	1742	172.39	36.24	.003	1282	135.3	34.74
alu	(.056)	144.40	.010	10198	586.57	146.12	.005	9633	129.32	143.91

80% Congestion for c7552 with different manufacturing variation cases										
case 1	(.086)	22.56	(.067)	1434	16.18	23.34	(.033)	1442	110.03	23.26
case 2	(.122)	16.00	(.096)	1166	71.21	16.54	(.065)	1229	105.95	16.44
case 3	(.086)	16.76	(.063)	1435	76.65	17.33	(.032)	1441	104.84	17.19
case 4	(.099)	27.91	(.079)	1315	20.38	28.82	(.046)	1314	102.52	28.52

algorithm LPECO reduces the negative slack better than the commercial design tool every case except the 70% alu).

In the cases where LPECO finds a timing feasible solution, it outperforms the commercial design tool in all metrics. The c_{area} metric is much better – in the 90% congestion case, it is half of the commercial tool's value, on average. In the same set of benchmarks, the c_{timing} is 12% less than the commercial tool, and the power is 5% less. This shows that the formulation in (2) is effective.

The difference between the initial power (p_{initial}) and optimized powers is much smaller than the change in the ECO timing and area measures. This shows that the ECO measures are important measures to quantify an ECO, as the resulting power change is near-negligible.

The designs with higher congestion fare better in the results because these designs have a longer wirelengths between gates. As the placement density grows, cell sizing is needed to recover timing under a tight placement.

Four additional manufacturing changes (Table 1) are run for benchmark c7552: case 1: +2% for all parameters; case 2: +5% for all parameters; case 3: +5% for nmos only; case 4: +5% for pmos parameters only. These results show that the changes on the pmos affect the timing more than changes on the nmos. In these cases, the negative slack cannot be corrected by either LPECO or the commercial design tool. However, the LPECO gives 43% less negative slack than the commercial tool, showing that the slack minimization formulation is effective.

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, and propose a method for performing ECO gate sizing. This leads to results that outperform a leading commercial design tool in the timing closure, and the resulting cost of the ECO for nearly all benchmark examples.

Further research will be made to extend this algorithm to the cases of layout-transparent changes such as v_t assignment and gate-length biasing, and to create a framework to run on the algorithm on large-scale designs.

6. ADDITIONAL AUTHORS

7. REFERENCES

[1] Available from http://www.opencores.org.

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

[3] M. Berkelaar, K. Eikland, and P. Notebaert. lp_solve 5.1. Available from http://lpsolve.sourceforge.net/5.1.

[4] J. Cong and M. Sarrafzadeh. Incremental physical design. In *Proc. Int. Conf. Physical Design*, page 92. ACM, 2000.

[5] O. Coudert, J. Cong, S. Malik, and M. Sarrafzadeh. Incremental cad. In *Proc. Int. Conf. Computer-Aided Design*, pages 236–244, 2000.

[6] A. Kahng and S. Mantik. On mismatches between incremental optimizers and instance perturbations in physical design tools. In *Proc. Int. Conf. Computer-Aided Design*, page 22. IEEE Press, 2000.