Matching-Based Minimum-Cost Spare Cell Selection for Design Changes

Iris Hui-Ru Jiang
Dept. of Electronics Engineering
National Chiao Tung University
Hsinchu 30010, Taiwan
huiru.jiang@gmail.com

Hua-Yu Chang
Freelance
Taipei, Taiwan
huayu.chang@gmail.com

Liang-Gi Chang and Huang-Bi Hung
Institute of Electronics
National Chiao Tung University
Hsinchu 30010, Taiwan
{lgchang, penn}.ee96g@nctu.edu.tw

ABSTRACT
Metal-only ECO realizes the last-minute design changes by revising the photomasks of metal layers only. This task is challenging because the pre-injected spare cells are limited both in number and in cell types. This paper proposes a matching-based ECO synthesizer, named ECOS, that correctly implements the incremental design changes using the available spare cells as well as tries to reduce the prohibitive photomask cost at the same time. The experiments are conducted on five industrial testcases. ECOS uses less photomask costs to complete design changes for all cases than the direct method that transforms the widely-used hand-editing procedure into an automatic one.

Categories and Subject Descriptors
B.7.2 [**INTEGRATED CIRCUITS**]: Design Aids – *Placement and Routing*

General Terms
Algorithms, Design

Keywords
ECO, spare cells, resynthesis, physical synthesis, matching

1. INTRODUCTION
Engineering change order (ECO) is a process that implements incremental design changes [1]. These modifications may result from functionality debugging, timing improvement, and specification revision. The later stage where ECO is performed, the less resources are available, and the greater challenges can be met. After the base layers (placement) are frozen during the design cycle, ECO not only can shorten design time by avoiding rebuilding the design from scratch but also can reduce fabrication time by manufacturing the base-layer photomasks in advance. After the first silicon chips are produced, ECO can save the prohibitive photomask cost by reusing the base-layer part in the next tape-out. Modifying only a few photomasks of metal layers to realize design changes is referred to as metal-only ECO. To facilitate metal-only ECO, a design is sprinkled with unused (spare) cells at placement, and their inputs are tied to either logic high or low to prevent floating signal. ECO is then performed by rewiring the inputs and outputs of spare cells.

DAC'09, July 26-31, 2009, San Francisco, California, USA

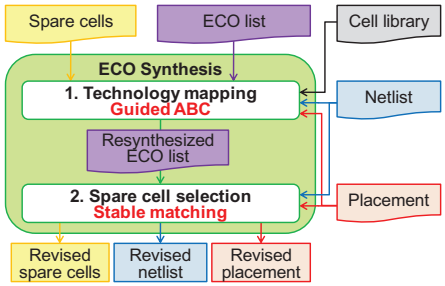

Figure 1. The overview of ECOS.

Good metal-only ECO relies on the following three techniques:

1. Sufficient and evenly sprinkled spare cells can accommodate design changes at all possible locations [2][3][4].
2. A good incremental eco-router can handle tremendous obstacles & design rules and complete routing with minimum changes [2][3][4][5].
3. A powerful ECO synthesizer can fulfill the design changes on functionality and/or timing by including the physical information into logic synthesis wisely and modeling the impact on the photomask cost when the selected spare cells deviate from the ideal locations [2][3].

For timing ECO, [6] proposed a technology-remapping technique based on dynamic programming. [7] fixed the input slew, output loading, and delay violations by inserting buffers (spare cells). However, these methods might not easily be extended for functional ECO. On the other hand, for functional ECO, [8] remapped the expected cells with available spare ones whose inputs could be inserted with constant values (logic high/low). However, this method did not consider the issues of non-tree type cells (e.g., multiplexors, full-adders), freeing up unused cells, and minimizing the photomask cost. It can hardly handle timing ECO.

To overcome the aforementioned drawbacks, this paper proposes a matching-based ECO synthesizer, named ECOS, to complete the functional changes with the minimum photomask cost. As shown in Figure 1, given the list of functional changes, the original netlist and placement, and the available spare cells, ECOS first resynthesizes the given ECO list using affordable spare cell types with geometry proximity consideration. Then, each instance in the resynthesized list is replaced by an adequate spare cell based on stable matching, as well as the related nets are reconnected. Moreover, the induced unobservable cells can be freed up for later

ECO runs. The objective of spare cell selection is to minimize the photomask cost of metal layers. Without loss of generality, this cost is modeled by the summation of the half-perimeter bounding box (HPBB) of each net in the revised design. This cost model benefits short interconnect delay, thus readily extending to timing ECO. Afterwards, formal equivalence checking can be performed to verify whether the revised design matches the revised functionality. ECOS has the following distinguished features:

1. It handles non-tree type spare cells and ECO functions.
2. It considers constant value insertion for spare cells.
3. It recycles freed-up cells for subsequent ECO runs.
4. It integrates physical information into resynthesis.
5. It solves the competition among spare cells.
6. It minimizes the photomask cost (also benefits timing).
7. It can readily extend to timing ECO.
8. It easily collaborates with existing synthesizers.

To demonstrate the effectiveness, we automate the common hand-editing ECO flow and then conduct the experiments on five industrial testcases. The results show that ECOS is promising.

2. PRELIMINARIES AND PROBLEM FORMULATION

2.1 Common ECO Synthesis Flow

Metal-only ECO is commonly performed by hand-editing the netlist [1]. However, this ad hoc method is time-consuming and resource intensive because the design related files have to be searched and edited many times during the whole process. Figure 2(a) shows an example design with two inputs, two outputs, and four logic cells. Spare cells include two AND and one NOT cells. The placement is also illustrated. For simplicity, the area of each cell in this example is 0, and all pins are located at the same point.

The photomask cost of metal layers depends on how complicated the whole routing could be. Hence, the photomask cost of each net is modeled by the half-perimeter bounding box (HPBB) over its related pins; the total cost of a design is the sum of HPBBs over all nets. For the design in Figure 2(a), we have its total cost:

HPBB(net1) + HPBB(net2) + HPBB(net3) + HPBB(in1) + HPBB(in2) + HPBB(out1) = 6,000 + 1,000 + 3,000 + 5,000 + 5,000 + 2,000 = 22,000.

Assume the functional ECO is to replace the functionality of cell U3 by AND. We have two options: spare cells SPARE1 and SPARE2. We prefer SPARE2 because of the better proximity to cells U1, U2, U4 and output out2. Moreover, in the revised design, the output of cell U3 becomes unused. Hence, the inputs of U3 can be connected to constant values instead. This kind of freed-up cells can be gathered and reused at subsequent ECO runs. The revised design can be improved as Figure 2(b). The total cost becomes 20,000.

Although not shown here, a spare cell can be used to implement more than one function, e.g., a two-input NAND cell can be a NOT cell by inserting a logic high to one input. The constant insertion can maximize the capability of each spare cell. Please note that the cell types of spare cells in most of cases do not directly match the ECO functionality. We need to translate the ECO functionality into pieces, and realize each piece by a spare cell. When the size of the ECO list is large and the resource of spare cells is limited, the ad hoc method would be time-consuming and may fail due to the competition among ECOs.

2.2 Problem Formulation

This paper solves the following metal-only ECO problem.

Minimum-Cost Spare Cell Selection for Functional ECO:
Given the original netlist, placement, a set of spare cells, and a list of functional changes, complete the ECO list using the available spare cells, create the revised netlist with the minimum cost, and generate the revised set of spare cells.

3. MATCHING-BASED ECO SYNTHESIS

This paper develops a matching-based ECO synthesizer, named ECOS, to solve the problem formulated in Section 2.2. Figure 1 details ECOS' inputs/outputs and summarizes its two steps.

3.1 Technology Mapping: Guided ABC

The first step of ECOS performs technology mapping to resynthesize the given ECO list using the available spare cell types. After this step, a resynthesized ECO list is produced. We build our synthesizer based on the well-established environment, ABC [9]. Basically, ABC performs optimal-delay DAG-based technology mapping, while guided ABC uses the physical information of spare cells to direct technology mapping. The cell library is modified for each ECO. Each spare cell corresponds to one library cell; its cell area reflects the cost when it is selected for this ECO, while its cell delay is set to zero. Doing so can trigger ABC to perform area-recovery since each possible mapping has the same delay.

As mentioned in Section 2.1, the cells that become unobservable after ECO are freed up and included in the revised set of spare cells for later ECO runs. They cannot affect the costs of their input and output nets after being recycled. In Figure 2(a), cell U3 does not affect the costs of net1, net2, and net3 after ECO1 is applied. After cell U3 in Figure 2(b) is freed up, the HPBBs of touched nets will be:

HPBB(net1) = 3,000; HPBB(net2) = 0; HPBB(net3) = 0.

Moreover, we can compute the HPBB of each ECO by finding the bounding box over all active pins in its related nets, e.g., we have

HPBB(ECO1)
= HPBB(net1, net2, net3) = HPBB(U1, out2, U2, U4) = 6,000.

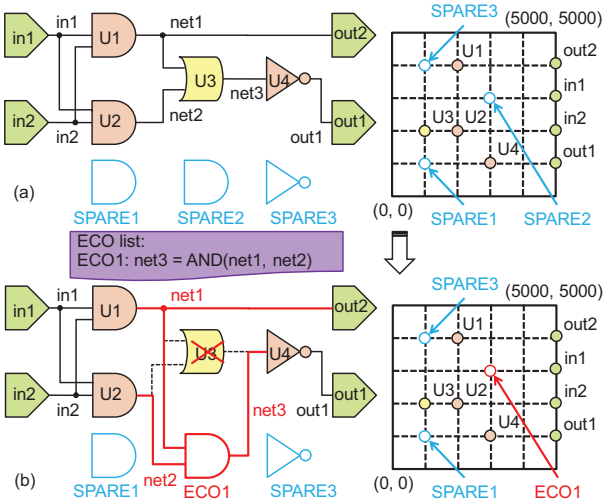

Figure 2. (a) The original design. (b) The revised design.

StableMatching(M, W)
1. Initialize all $m \in M$ and $w \in W$ to free
2. **while** ∃ free man m with a woman w to propose to **do**
3. w = m's highest ranked such woman
4. **if** w is free **then**
5. (m, w) become engaged
6. **else** // some pair (m', w) already exists
7. **if** w prefers m to m' **then**
8. (m, w) become engaged
8. m' becomes free

Figure 3. The stable matching algorithm [10].

The value can be viewed as the lower bound of the total cost induced by an ECO. The cost of selecting a spare cell for a given ECO can be computed accordingly. For ECO1, the costs of SPARE1, SPARE2, and SPARE3 are 7,000, 6,000, and 7,000, respectively. Hence, the cell library for ECO1 contains one SPARE1 cell of function/area/delay AND/7,000/0, one SPARE2 cell of function/area/delay AND/6,000/0, and one SPARE3 cell of function/area/delay NOT/7,000/0. Based on the modified cell library, guided ABC can generate the best choice for each ECO. For example, for ECO1, SPARE1 and SPARE2 have the same delay (zero), so the cell of smaller area is selected, i.e., ECO1 is resynthesized as a SPARE2 cell. Then, the resynthesized ECO list contains only the cell types of available spare cells. Sometimes, an ECO in the original ECO list may be converted into several cells. Moreover, DAG-based technology mapping cannot handle non-tree type spare cells and ECO functions. We resort this problem to ROBDDs. If the spare cell types are a mixture of only multiplexors (MUX) and inverters (NOT), the ECO list will be transformed to ROBDDs first; these ROBDDs are then simplified and converted to MUX/NOT cells. In addition, constant value insertion can naturally be implemented in technology mapping, thus maximizing the capability of spare cells. It can be seen that the guidance made for ABC indeed can also easily be built in other existing logic synthesizers provided by EDA vendors.

3.2 Spare Cell Selection: Stable Matching

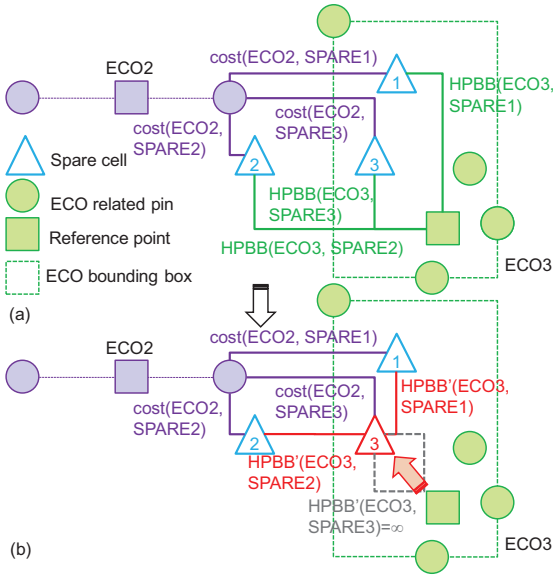

Figure 4. The reference points and bounding boxes: (a) Before engagement. (b) After engagement.

Although guided ABC considers physical information of spare cells, it cannot handle the competition among ECOs. If a spare cell is the best choice of several ECOs, guided ABC duplicates it for these ECOs. To solve this problem, we do not select spare cells directly in guided ABC, but defer the decision making to step 2. With the global view of costs, deferred decision making at step 2 may lead to good results.

3.2.1 The Stable Marriage Problem

Stable Marriage:
Given a set of n men and m women, marry them off in pairs after each man has ranked the women in order of preference and each women has done likewise such that no pair of man and woman who would both rather have each other than their current partners. If there are no such pairs, all the marriages are stable.

This paper reduces spare cell selection to the stable marriage problem, handling the competition among candidates in nature. Gale and Shapley proposed a stable matching algorithm listed in Figure 3, which is male-optimal [10].

3.2.2 The Reduction

Due to male-optimality, each ECO in the resynthesized list is modeled as a man, while each spare cell is modeled as a woman. The preference is defined by the added cost resulting from assigning a spare cell to an ECO. The less added cost, the more preference. The added cost contains the difference made on the existing nets and the induced cost on the newly created nets. (Please note that the added cost is different from the cost used in guided ABC.) If there are no newly created nets among the ECOs in the resynthesized list, the preference order can be determined directly, e.g., ECO1 in Figure 2(a) has the following preferences:

pref(ECO1, SPARE1) = cost(ECO1, SPARE1) = 1,000;
pref(ECO1, SPARE2) = cost(ECO1, SPARE2) = 0.

Thus, ECO1 prefers SPARE2 and proposes to SPARE2. SPARE2 then accepts, and the stable matching is found. If no ECOs in the resynthesized list are connected each other, the stable matching algorithm leads to good results for all ECOs. However, when there are internal connections among some ECOs, the spare cell selection for ECO would affect each other. This case may occur when an ECO is resynthesized into several ECOs; some newly created nets connect among ECO cells only. Because the designated spare cells for ECOs have not been decided yet, the cost cannot be determined. To estimate the induced cost on the newly created nets, the reference point of each ECO is introduced to represent a good location for spare cell selection. It is initially set as the average x- and y-coordinates over all pins on its related nets. For example, the reference point of ECO1 in Figure 2(a) is at:

x(ECO1) = 3,000; y(ECO1) = 2,750.

With setting the reference points, we can compute the induced cost on newly created nets and then rank the preference between ECOs and spare cells. For example, Figure 4(a) depicts the reference points and bounding boxes of ECO2 and ECO3. Assume a newly created net connecting ECO2 and ECO3. The preference of ECO2 proposing to SPARE1 consists of the added cost between SPARE1 and the existing nets of ECO2 and the estimated cost of the net between ECO2 and ECO3.

pref(ECO2, SPARE1) =
cost(ECO2, SPARE1) + HPBB(ECO3, SPARE1).

As the stable matching algorithm progresses, once an ECO is engaged to a spare cell, its reference point is updated to the real location of the engaged spare cell. As shown in Figure 4(b), after ECO3 proposes to SPARE3 and gets accepted, the estimated cost of the net between ECO2 and ECO3 is updated as follows.

HPBB'(ECO3, SPARE1) = HPBB(SPARE3, SPARE1).
HPBB'(ECO3, SPARE2) = HPBB(SPARE3, SPARE2).
HPBB'(ECO3, SPARE3) = ∞.

The estimated cost between ECO3 and SPARE3 is set to a large value rather than 0 to prevent ECO2 from proposing to SPARE3. Doing so can guarantee that the method is stable and always has a solution. Spare cell selection follows the stable matching algorithm, except lines 5 and 8 in Figure 3. The reference point of an ECO is updated when it is engaged. The preference ranking related to this ECO is updated accordingly; this update would not affect the processed proposals to maintain stability.

3.2.3 The Complexity of Spare Cell Selection

The preference can be computed during step 1. Ranking n ECOs and m spare cells can be done in $O(nm\lg m + mn\lg n)$ time. The worst case of the stable matching algorithm is $O(nm\lg n)$. Hence, the overall time complexity of spare cell selection is $O(nm(\lg m+\lg n)) = O(nm\lg m)$ since n is less than or equal to m. Because only the spare cells of the same type of an ECO in the resynthesized list can be matched, the expected number of possible proposals is far less than nm, thus the practical complexity is quite lower than the worse case.

4. EXTENSION TO TIMING ECO

For timing ECO, we may insert buffers into timing critical paths. These buffers can be viewed as functional ECO, and ECOS naturally fixes timing by minimizing the related HPBBs.

5. EXPERIMENTAL RESULTS

We implemented our algorithm in C++ language and executed the program on a PC with an Intel® Core™2 CPU T7400 of 2.16 GHz frequency and 2 GB memory. Totally five industrial testcases are used. The first three use a general cell library with basic and complex logic cells, while the last two use a cell library with only multiplexors and inverters. The left part of Table 1 lists the number of pins, the number of cells, the number of nets, and the number of spare cells of testcases. The netlist and placement of the original design are described in DEF format, while the ECO list is specified in VERILOG format. We automated the common ECO method as the direct method. It searches for a spare cell with the required functionality within the bounding box of each ECO. If none, blind ABC resynthesizes each ECO with the available spare cell types (the area & delay of each spare cell are 0). Then, each ECO in the resynthesized list is directly mapped to one spare cell of the same type inside its bounding box. If failed, it is alternatively mapped to someone else outside.

Table 1. Statistics on testcases and on ECO.

	Statistics				#ECOs			
Case	#Pin	#Cell	#Net	#Spare	#Req	#FR	Blind ABC	Guided ABC
testcase1	483	28,591	28,705	350	7	7	16	11
testcase2	483	28,591	28,705	2,300	49	51	295	153
testcase3	483	28,591	28,705	2,300	94	121	313	162
testcase4	33	198	181	40	3	3	5	4
testcase5	30	938	850	100	4	4	6	5

#Req: number of required ECOs; #FR: number of freed up cells

The right part of Table 1 lists the number of the given ECO list, the number of freed-up cells and the sizes of resynthesized ECO lists for blind & guided ABC. The number of freed-up cells could be greater than the size of the ECO list when the freed up cell has multiple outputs. Guided ABC generates a much fewer ECOs in the resynthesized list than blind ABC. The smaller resynthesized ECO list may result in the smaller overlapped bounding boxes and then lead to the lower cost. Table 2 compares the total cost, the ECO related HPBB before & after ECO, and CPU times. The original values are listed here for reference. Ratio means the cost normalized to the direct method. The direct method incurs average 7.66% degradation on the total cost. Considering the HPBB of ECO (which can be viewed as the lower bound of the cost induced by ECO), ECOS on average outperforms the direct method by 47.09% on the total cost and always produces the better results. In addition, compared with the timing-consuming manual method, the automatic methods are efficient.

6. REFERENCES

[1] S. Golson. The human eco compiler. In *Proc. SNUG*, pages 1-57, 2004.
[2] Synopsys, Inc. http://www.synopsys.com/.
[3] Cadence Design System. http://www.cadence.com/.
[4] Magma Design Automation. http://www.magma-da.com/.
[5] J.-Y. Li and Y.-L. Li. An efficient tile-based ECO router with routing graph reduction and enhanced global routing flow. In *Proc. ISPD*, pages 7-13, 2005.
[6] Y.-P. Chen, J.-W. Fang and Y.-W. Chang. ECO timing optimization using spare cells and technology remapping. In *Proc. ICCAD*, pages 530-535, 2007.
[7] C.-P. Lu, M. C.-T. Chao, C.-H. Lo, and C.-W. Chang. A metal-only-ECO solver for input slew and and output loading violations. In *Proc. ISPD*, pages 191-198, 2009.
[8] Y.-M. Kuo, Y.-T. Chang, S.-C. Chang and M. Marek-Sadowska. Engineering change using spare cells with constant insertion. In *Proc. ICCAD*, pages 544-547, 2007.
[9] ABC: A system for sequential synthesis and verification. http://www.eecs.berkeley.edu/~alanmi/abc/.
[10] D. Gale and L. S. Shapley. College admissions and the stability of marriage. *American Mathematical Monthly*, vol. 69, pages 9-14, 1962.

Table 2. Comparison on total cost, HPBB of ECO, and CPU time.

	Total cost			HPBB of ECO			CPU time (sec)	
	Before ECO	Direct	**ECOS**	Before ECO	Direct	**ECOS**	Direct	**ECOS**
testcase1	4,049,536,290	4,062,343,750	**4,051,305,350**	8,124,160	24,336,840	**12,621,060**	0.35	**0.51**
testcase2	4,142,631,960	4,414,703,560	**4,336,177,520**	78,648,240	301,202,720	**194,842,960**	1.20	**5.41**
testcase3	4,142,631,960	4,640,619,480	**4,553,033,540**	166,895,860	572,380,280	**466,332,680**	1.63	**9.38**
testcase4	2,310,270	2,652,430	**2,323,150**	114,800	501,200	**164,080**	0.10	**0.14**
testcase5	12,945,900	14,098,100	**12,989,860**	413,250	1,677,700	**567,220**	0.15	**0.17**
Ratio	0.9234	1.0000	**0.9516**	0.2724	1.0000	**0.5291**	-	**-**