Synthesis and Utilization of Standard Cells Amenable to Gear Ratio of Gate-Metal Pitches for Improving Pin Accessibility

Jooyeon Jeong, Sehyeon Chung, Kyeongrok Jo and Taewhan Kim
Department of Electrical and Computer Engineering
Seoul National University
Email : {jooyeon, shchung, jkr2100, tkim}@snucad.snu.ac.kr

***Abstract*—Traditionally, the synthesis of standard cells invariably assumes that the gear ratio (GR) between the gate poly pitch in the cells and the metal pitch of the first vertical metal layer (to be used for routing) over the gate poly is 1:1 for chip implementation. However, the scaling trend in sub-10nm node CMOS designs is that GR is changing from 1:1 to 3:2 or 4:3, which means the number and location of pin access points vary depending on the cell placement location, thereby causing hard-to-pin-access if the pin access points were aligned on the off-track routing pattern. This work overcomes the pin inaccessibility problem caused by non-1:1 GR in chip implementation. Precisely, we propose a non-1:1 GR aware DTCO (design and technology co-optimization) flow to generate cells with pin patterns that are best suited to the implementation of target design. To this end, we propose two new tasks to be installed in our DTCO framework: (1) from the existing cells optimized for 1:1 GR, we *relocate their pin patterns amenable to non-1:1 GR, so that a maximal pin accessibility should be achieved*; (2) we incrementally update the pin patterns of the cell instances with routing failures due to pin inaccessibility in the course of the DTCO iterations to *produce the cells with best fitted pin patterns to the implementation of target design*. We formulate task 1 into a problem instance of dynamic programming to find an optimal solution of pin positions, considering design rule and access conflict constraints while we solve task 2 by devising an assessment function on the pin accessibility enhanced by pin pattern extension to find out the most suitable direction for the extension. In the meantime, through experiments with benchmark circuits, it is shown that our DTCO methodology optimizing pin patterns amenable to non-1:1 GR is able to produce chip implementations with on average 5.88× fewer routing failures at no additional wirelength, timing, and power cost.**

I. Introduction

The ratio between contacted poly pitch (CPP) and M2 pitch (M2P) is commonly referred to as *gear ratio* (GR). Traditionally, GR has been 1:1 i.e., the same pitch for gate poly and M2. However, GR of $\gamma_1 : \gamma_2$ where $\gamma_1 > \gamma_2$ is preferred to avoid using an additional metal routing layer [1]. For example, according to ASAP PDK at 7nm process [2], CPP is 54nm and M2P is 36nm, thus GR = CPP:M2P = 3:2. Fig. 1 shows, for each of GR = 1:1, 3:2, and 4:3, the vertical alignment of M2 tracks (green) over gate polys (red) for various cell placement instances where ϕ indicates the cell *offset* value of the leftmost gate poly in cell placement to its first M2 track on the right side. Clearly, shrinking the M2 pitch (i.e., using more M2 tracks), as shown in Fig. 1, can improve the routability.

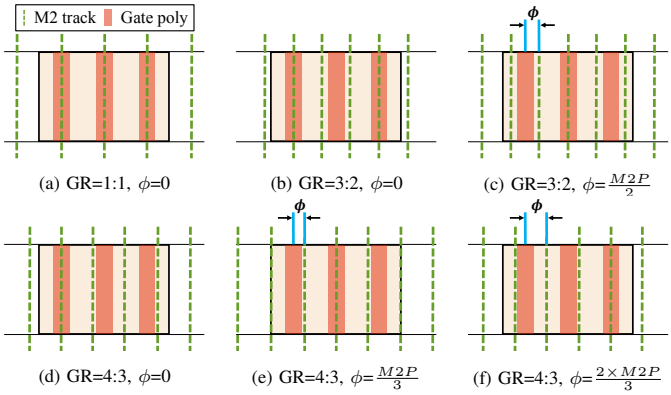
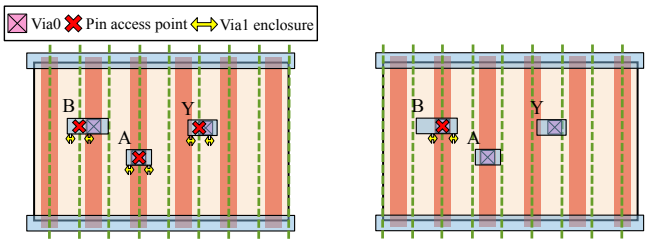


Fig. 1. Cell instance or gate poly alignment to the M2 tracks for GR=1:1, 3:2, and 4:3 where ϕ indicates the cell offset value of the leftmost gate poly in cell placement to its first M2 track on the right side.

For non-1:1 GR, placements of cell instances to different locations may induce different M2 track alignments over the gate polys of the instances. This implies that for non-1:1 GR, the number and position of pin access points on cell instances can vary depending on the cell placement location, as illustrated in Fig. 2, in which an instance of AND-gate cell with A, B, and Y as I/O pins can have an access point for every pin if the instance were to be placed at a location with offset ϕ = 0 as shown in Fig. 2(a). On the other hand, it loses two access points if it were placed to a location with $\phi = \frac{M2P}{2}$ as shown in Fig. 2(b).

To our knowledge, only a few works have addressed on the problem of solving the pin inaccessibility caused by the non-1:1 GR. The work in [3] introduced a concept of *floating* M2 segments to defer in-cell routing with the M2 segments until cell placement is completed. Then, it moved the floating M2 segments to the left or right to make M2 on-track alignment. The serious problem of this method is that the cell layouts are incomplete. Thus, the cell characterization whose characterized values, such as delay and power measures, are to be used in physical design cannot be processed or inaccurate. Furthermore, it wastes in-cell routing resource since the cell layout should prepare in-cell connections to *both* the left and

978-3-9819263-7-8/DATE23/© 2023 EDAA



(a) Placement of AND-gate cell with $\phi = 0$, resulting in a total of 3 access points to M2

(b) Placement of AND-gate cell in (a) with $\phi = \frac{M2P}{2}$, resulting in a total of 1 access point to M2

Fig. 2. Changes of the number of access points to M2, depending on the placement of cell instance.

right of the floating segments through exactly one of them will be actually needed. The work in [4] demonstrated the benefit of using more routing metal resources by adopting 3:2 GR as opposed to 1:1 GR. It generated cells from the existing cells by simply widening CPP by 7% and shortening M2P by 29% to make GR = 3:2. Then, they performed cell placement and net routing for an SoC CPU implementation and showed that using 3:2 GR achieved 17% higher performance and 8% lower power consumption. However, this did not take into account the degradation of routability induced by offset variation. On the other hand, [5] observed the pin inaccessibility by M2 track misalignment in placement, and proposed, during the detailed placement stage, to shift the cell instances to the left or right to maximize the total number of pin access points. However, shifting cells may affect the other important parameters, such as timing and wirelength, which have already been optimized.

We prevent the possible loss of pin access points and routing failures from M2 track misalignment over the pin patterns on cell instances during the post-routing stage as well as the post-placement stage. Our key idea is that, unlike the previous approaches, where they apply cell shifting or use cells with longer pin patterns than necessary, we iteratively explore pin patterns to find the best fit to the target implementation through building up a non-1:1 GR aware DTCO framework.

II. Algorithms for Exploring and Synthesizing Cells Amenable to non-1:1 GR

A. The Proposed DTCO Framework

Fig. 3 shows the framework, called DTCO-pinOpt, of our DTCO for optimizing cell pin patterns considering the cell offset value varied by the cell placement position. The inputs to the DTCO framework are cells, $\{C_i\}$, optimized for 1:1 GR, an initial target design D_0, and target GR = γ_1:γ_2. The two main tasks in our DTCO framework are (*task* I) generating, for each offset value ϕ from γ_1:γ_2 GR, new cells, $\{C_i^\phi\}$, from $\{C_i\}$ with *optimal pin accessibility location on the pin patterns* of minimal length and replacing the cell instances of $\{C_i\}$ with ϕ in placement with $\{C_i^\phi\}$ and (*task* II) *minimally extending pin patterns of the cell instances with routing failure* to increase the pin accessibility in ECO routing. We iterate *task* II followed by ECO routing until the number of DRVs

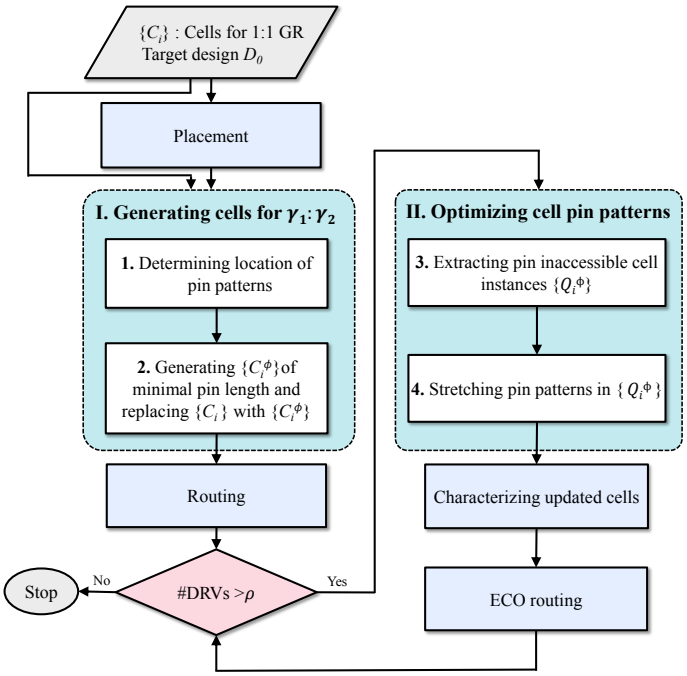


Fig. 3. Our DTCO framework DTCO-pinOpt for optimizing cell pin patterns considering the cell offset value (i.e., ϕ) varied by the cell placement position.

(design rule violations) caused by pin inaccessibility in routing does not exceed the limit ρ. In the following subsections, we provide the details on *tasks* I and II.

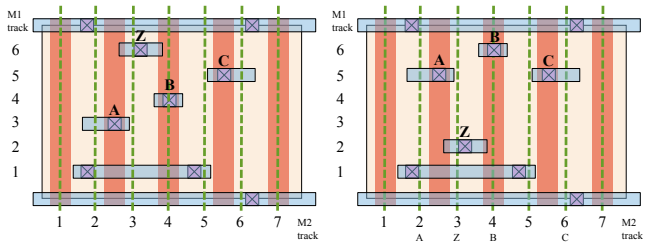
B. Determining Location of Pin Patterns

This step is to determine the location of every pin pattern of minimal length in a cell with offset ϕ. For example, the cell in Fig. 4(a) contains three input pins A, B, C, and one output pin Z such that access patterns of A, B, C, and Z are located at (2,3), (4,4), (6,5), and (3,6), respectively where the x-coordinate corresponds to M2 track index and the y-coordinate corresponds to M1 track index. The pin pattern distribution in Fig. 4(a) shows that it is much hard to access pin B due to the close blockage of other pins. On the other hand, Fig. 4(b) shows the same cell as that in Fig. 4(a), but it has a pin pattern distribution with better accessibility.

As illustrated in Figs. 4(a) and (b), the problem of determining pin pattern location is to find an M1 track (i.e., y-coordinate) for each pin pattern of the minimal length so that overall pin accessibility should be maximized. (We assume that the technology node used in this work allows gate contact to be placed directly on the active region [6], which means a pin pattern can be placed at any vertical location of its poly gate contact over P-active, N-active, or the middle region between P/N-actives.)

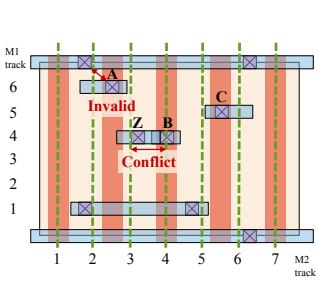
We determine the pin pattern location in a way to satisfy the following two objectives.

$\mathcal{O}_1$: The distance between the locations at which two horizontally adjacent pin patterns are placed should be as far as possible to reduce the risk of routing conflict.

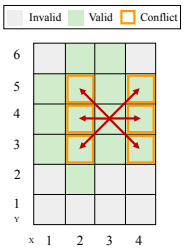


(a) A pin pattern distribution, providing low pin accessibility

(b) A pin pattern distribution, providing high pin accessibility



(c) Invalid and conflict locations



(d) 2-D table with (x, y)-coordinate, describing location classification

Fig. 4. Example showing the effect of pin pattern distribution on pin accessibility and classification of pin locations.

$\mathcal{O}_2$: Input pins prefer the M1 track locations that are on the middle region over cells to minimize the delay skew between their PMOS (i.e., upper) and NMOS (i.e., lower) transistors.

Definitions and notations: We call a location (x, y) *invalid* if forming a pin pattern on (x, y) causes a via-spacing rule or metal spacing rule violation because of intra-cell wires except for pin patterns, and *valid*, otherwise. In addition, we call two locations *conflict* if forming two pin patterns causes such DRVs. Fig. 4(c) shows an example of invalid and conflict locations for the pins in Fig. 4(a).

In a cell with offset ϕ, let $X = \{1, 2, \cdots\}$ and $Y = \{1, 2, \cdots\}$ denote an ordered enumeration of the M2 track index list corresponding to the pin patterns of the pins in the cell and of the M1 track index list on the cell that are sorted according to the increasing order of x- and y-coordinate, respectively. (For a pin pattern on multiple M2 tracks, we choose the middle M2 track as x-coordinate.) Finally, let $V(i)$ denote the set of valid locations of the pin pattern corresponding to pin index $i \in X$. For example, from Figs. 4(a) or (b), $V(1) = \{(1,3), (1,4), (1,5)\}$, $V(2) = \{(2,2), (2,3), (2,4), (2,5), (2,6)\}$, and so on. Then, Fig. 4(d) shows the location classification of the pins in Figs. 4(a) or (b).

Problem formulation: We formulate the problem of finding pin pattern locations for all pins in a cell into a problem instance of dynamic programming. Dynamic programming is an algorithm that solves big problems by dividing them into small ones. We can increase the speed by saving answers from small ones and using them to solve the original problem. To this end, first, we define a pin location cost function, C_{loc}, which takes into account objectives $\mathcal{O}_1$ and $\mathcal{O}_2$ when placing pin patterns to two locations $v_1 = (i, j_1)$ and $v_2 = (i{+}1, j_2)$ where $v_1 \in V(i)$, $v_2 \in V(i{+}1)$, and $i \in X$:

$$C_{loc}(v_1, v_2) = (1 - conflict(v_1, v_2)) \cdot (c_{dist}(v_1, v_2) + c_{track}(v_2)) \quad (1)$$

where $conflict(v_1, v_2) = 1$ if v_1 is conflict with v_2, and 0, otherwise,

$$c_{dist}(v_1, v_2) = HPWL(v_1, v_2), \quad (2)$$

in which $HPWL(v_1, v_2)$ is the half-parameter wire length of the bounding box of v_1 and v_2, reflecting $\mathcal{O}_1$, in terms of M1P (M1 pitch) units, and

$$c_{track}(v_2) = prefer(v_2) \cdot \frac{|Y|}{\alpha}, \quad (3)$$

in which $prefer(v_2) = 1$ if v_2 is an input pin pattern and on an M2 track in the middle region over the cell, reflecting $\mathcal{O}_2$, and $\frac{1}{2}$, otherwise. $|Y|$ is an index number of M1 tracks passing over the cell and α, $0 \leq \alpha \leq |Y|$ ($\alpha \neq 0$) is a control parameter to balance $\mathcal{O}_1$ and $\mathcal{O}_2$.

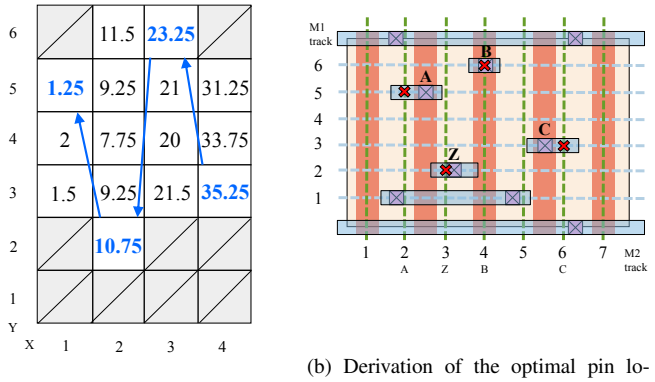
We use $R[i, j]$, $i \in X$ and $j \in V(i)$, to measure the goodness, obeying $\mathcal{O}_1$ and $\mathcal{O}_2$, of pin pattern location of all pins indexed by $\{1, 2, \cdots, i\} \subseteq X$ with the constraint of the pin pattern of index i being placed on location (i, j). We define basis $R[1, j]$ for $j \in V(1)$ followed by the recurrence relation of $R[i, j]$ for $i \in X$, $i \neq 1$ and $k \in V(i-1)$:

$$R[i,j] = \begin{cases} prefer(v(1,j)) \cdot \frac{|Y|}{\alpha}, & i = 1 \\ & j \in V(1), \\ max\{R[i-1,k] + C_{loc}(v_1, v_2)\}, & i \neq 1 \\ & k \in V(i-1), \end{cases} \quad (4)$$

in which $v_1 = (i-1, k)$ and $v_2 = (i, j)$, from which we derive the pin pattern locations with the maximal goodness quantity:

$$C_{loc}^{opt} = max\{R[|X|, j], \quad (|X|, j) \in V(|X|)\}. \quad (5)$$

Example: Fig. 5(a) shows a step-by-step computation of $R[i, j]$ values for the cell in Fig. 4(b) with $\alpha = 2$. The $R[i, -]$ values are computed column-by-column, from the leftmost to the right. The values in the leftmost column correspond to the basis. Then, $R[i, -]$ values in column index i are computed by *Eq.4* using $R[i-1, -]$ values. For example, $R[3, 5] = max\{21, 17, 13.5, 9.25, 11.5\} = 21$ and $R[4, 3] = max\{21.5, 27.5, 31, 35.25\} = 35.25$. The heavy lines indicate the backward tracking to find R[-,-] values (blue color) that produces $C_{loc}^{opt} = 35.25$. Fig. 5(b) shows the location of pin patterns of minimal length corresponding to the optimal solution in Fig. 5(a). (Note that we used six M1 tracks for convenience in calculation, and used the design rules and M1P values in ASAP 7nm PDK [2] in which M1P = 18nm except



(a) Recursive cost computation in *Eq.4* for reallocating pin contacts

(b) Derivation of the optimal pin location from the recursive solution obtained in (a)

Fig. 5. A step-by-step evaluation based on dynamic programming for determining the location of pin patterns that maximizes C_{loc}^{opt} in *Eq.5*.

M1P between tracks 2 and 3 and between tracks 4 and 5 are 27nm to classify *valid*, *invalid* and *conflict* locations.)

Time complexity: Since $|V(i)| \leq |Y|$, for every $i \in X$, the computation of $R[i,-]$ for each iteration is bounded by $O(|Y|^2)$. Thus, the total computation time spent by our proposed dynamic programming is bounded by $O(|X| \cdot |Y|^2)$ since $|X|$ (i.e., the number of pins) iterations are required to finally compute C_{loc}^{opt} in *Eq.5*.

C. Replacing $\{C_i\}$ Instances with $\{C_i^\phi\}$

At the post-placement stage, we replace cell instances of $\{C_i\}$ with ϕ by the corresponding cell instances of $\{C_i^\phi\}$ that has been obtained from the prior step of determining the location of pin patterns of minimal length. In addition, we perform the cell characterization for the cells in $\{C_i^\phi\}$.

D. Extracting Accessing Failure Cell Instances

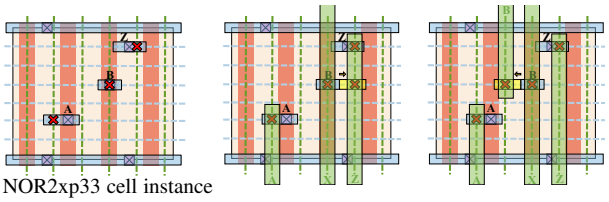
At each iteration, for the cell instances with routing failure by pin inaccessibility, we want to extend the M1 pin patterns in length so that the inaccessible pins each has one more access point. Let $\{C_i^\phi\}$ be the cell library that includes all cells produced so far and $\{I_i^\phi\}$ be the set of cell instances with the routing failure in the current DTCO iteration. Then, the pin pattern optimization task starts from extracting a cell subset $\{Q_j^\phi\} \subseteq \{I_i^\phi\}$ that requires *new cells* with optimal pin pattern extension for replacement in the current DTCO iteration before applying ECO routing.

We use a commercial placement and routing tool[1] to read the pin locations at which DRVs have occurred and collect the corresponding cells to set $\{Q_j^\phi\}$. (The DRVs include various pin inaccessibility issues like M2 short, end of line spacing, corner spacing, etc. [8].) This is a preparation step to see what cell instances should be updated to improve pin accessibility.

[1]We used Cadence Innovus [7].

E. Stretching Pin Pattern Length

This step generates new cells, $\{Q'^\phi_j\}$ of minimal-cost pin pattern extension for the cell instances in $\{Q_j^\phi\}$ and replace the instances in $\{Q_j^\phi\}$ by $\{Q'^\phi_j\}$, subsequently, characterizing the cells in $\{Q'^\phi_j\}$ and applying ECO routing.



(a) NOR2xp33 cell instance of $\phi = 0$ with access failure on pin B

(b) Extending pin B in (a) to the right

(c) Extending pin B in (a) to the left

Fig. 6. Two options of extending pin pattern. (a) A cell instance with access failure on B. (b) Extending to the right (i.e., $\lambda_B = 1$), still causing access failure on B. (c) Extending to the left (i.e., $\lambda_B = 0$), resolving access failure on B.

We can consider two options to extend a pin pattern that is not accessible in routing. One is extending it to the right to include one more access point and the other is to the left. We use notation $\lambda_p = 1$ to indicate the extension to the right of pin pattern of pin p and $\lambda_p = 0$ to the left, and $AP(\lambda_p)$ represents the access point newly added by λ_p.

For example, Fig. 6(a) shows a cell instance in placement which has an access failure on pin B. Figs. 6(b) and (c) show the pin extension with $\lambda_B = 1$ and $\lambda_B = 0$ of pin B in Fig. 6(a), respectively. Clearly, $\lambda_B = 0$ is a better choice for extension since $\lambda_B = 1$ still causes access failure by the blockage Z over $AP(\lambda_B = 1)$.

As illustrated in Fig. 6, we should determine the extension direction which maximally satisfies the following factors.

$\mathcal{F}_1$: The number of pin patterns on the M2 track where $AP(\lambda_p)$ resides should be as small as possible since those may collectively act as an obstacle in accessing $AP(\lambda_p)$.

$\mathcal{F}_2$: The distance between $AP(\lambda_p)$ and a pin pattern, if it exists, on the M2 track on which $AP(\lambda_p)$ resides should be as far as possible since the longer the distance is, the higher the possibility of accessing $AP(\lambda_p)$ is.

First, we define some notations used in our cost formulation of pin accessibility improvement by pin pattern extension.

$L_{dir}^{\lambda_p}$: It represents the list of access points that are on the M2 track (including the access points inducing a spacing rule violation) on which $AP(\lambda_p)$ resides, which are sorted according to the increasing and decreasing order of the values of y-coordinate, starting from $AP(\lambda_p)$ if dir = *upward* and dir = *downward*, respectively. For example, in Figs. 7(a) and (b), $L_{up}^{\lambda_C=1}$ = [B, A], $L_{down}^{\lambda_C=1}$ = [], $L_{up}^{\lambda_C=0}$ = [B], and $L_{down}^{\lambda_C=0}$ = [Y, D].

$D_{dir}^{\lambda_p}$: It represents the list of the track distances from $AP(\lambda_p)$ to the individual access points in $L_{dir}^{\lambda_p}$. For example, in Figs. 7(a) and (b), $D_{up}^{\lambda_C=1}$ = [1, 2], $D_{down}^{\lambda_C=1}$= [], $D_{up}^{\lambda_C=0}$ = [1], and $D_{down}^{\lambda_C=0}$ = [2, 3].

$Pr(p_i)$: It indicates the likelihood of using access point p_i. Assuming an equal probability of using the access points on the pin pattern, where $n_{ap}(p_i)$ is the number of access points on the pin pattern which includes p_i.

$$Pr(p_i) = \frac{1}{n_{ap}(p_i)} \tag{6}$$

$\overline{Pr}(p_i)$: It equals to 1 - $Pr(p_i)$.

We define a cost function, $C_{inacc}(\lambda_p, dir)$, that measures the badness of pin accessibility by pin extension:

$$C_{inacc}(\lambda_p, dir) = \frac{Pr(L_{dir}^{\lambda_p}[0])}{D_{dir}^{\lambda_p}[0]} + \frac{Pr(L_{dir}^{\lambda_p}[1]) \cdot \overline{Pr}(L_{dir}^{\lambda_p}[0])}{D_{dir}^{\lambda_p}[1]} + \cdots \tag{7}$$

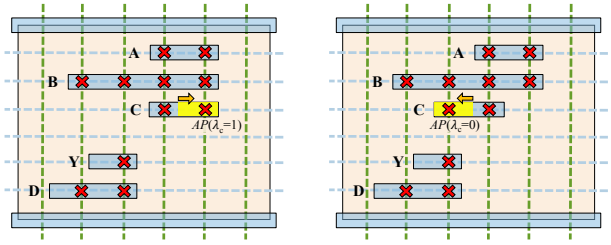
where the first term in the right equation expression corresponds to the inaccessibility cost of $AP(\lambda_p)$ when the closest access point (i.e., $L_{dir}^{\lambda_p}[0]$) becomes a blockage, the second term shows when the second closest one (i.e., $L_{dir}^{\lambda_p}[1]$) becomes a blockage, but the first closest one becomes no blockage, and so on.

Note that the numerator terms (i.e., $Pr(\cdot)$) in *Eq.7* takes into account factor $\mathcal{F}_1$ while the denominator terms (i.e., $D(\cdot)$) takes into account factor $\mathcal{F}_2$. Then, we determine the pin pattern extension direction of pin p by computing:

$$C_{inacc}^{opt} = min\{\sum_{k=\{0,1\}} C_{inacc}(1,k), \sum_{k=\{0,1\}} C_{inacc}(0,k)\} \tag{8}$$

where $k = 1$ for *dir* = *upward* and $k = 0$ for *dir* = *downward*.

Example: Fig. 7 shows an example illustrating our inaccessibility cost computation for extending a pin pattern. Fig. 7(a) computes the inaccessibility cost for extending to the right: $C_{inacc}(\lambda_C = 1, 1)$ = $\frac{\frac{1}{4}}{1} + \frac{\frac{1}{2}\times\frac{3}{4}}{2} = \frac{7}{16}$ and $C_{inacc}(\lambda_C = 1, 0)$ = 0. On the other hand Fig. 7(b) shows the cost for extending to the left: $C_{inacc}(\lambda_C = 0, 1)$ = $\frac{1}{4}$ and $C_{inacc}(\lambda_C = 0, 0)$ = $\frac{1}{2}$. Thus, C_{inacc}^{opt} = $min\{\frac{7}{16}, \frac{3}{4}\} = \frac{7}{16}$, which means extending the pin pattern of pin C to right is preferable.



(a) Extending C to the right (b) Extending C to the left

Fig. 7. Example to illustrate our accessibility cost computation for extending a pin pattern.

III. Experimental Results

A. Experiments Setup

We implemented our proposed method **DTCO-pinOpt** of pin pattern optimization amenable to non-1:1 GR based on dynamic programming with Python3 and used a Linux machine with AMD Ryzen 3970X CPU running at 2.2GHz with 128GB memory. Based on ASAP 7nm cells in [2] , we re-synthesized a total of 60 types of cell layouts. (Note that the cell library contains all kinds of cells, but to prevent the cell library from becoming large, we excluded some cells with different widths. We converted 2-D M1 routing to 1-D to focus on the problem caused by non 1-1 GR and there is no open cell library using 1-D M1 routing. And for securing enough M1 routing resources, we used the middle-of-line layers(i.e., LISD and LIG) as routing resources.) We set GR = CPP:M2P = 3:2 in our experiments, which was in fact assumed and used in ASAP 7nm PDK and created two versions of pin accessibility-optimized cells of the same type, one for offset ϕ=0, the other for $\phi = \frac{M2P}{2}$.

We checked all the synthesized cells with DRC (design rule check), LVS (layout versus schematic), and PEX by using Mentor Calibre [9] tool and characterized them by using Synopsys SiliconSmart [10]. We tested our pin pattern optimized standard cells on implementing the OpenCores [11] benchmark circuits for GR = 3:2 using the CPP and M2P values in [2], which are 54nm and 36nm, respectively. For each benchmark, we set chip utilization to 0.7(The value of utilization does not matter to the experiments.), and tried to optimize timing. In case a large value of negative timing slack still exists even when timing optimization consistently was applied to the circuits, we adjusted the clock frequency parameter and optimized timing to clean the negative timing slacks in circuits as many as possible. In addition, in experiments, we set M2 to the lowest metal layer for routing.

We prepared three DTCO flows to assess the effectiveness of our proposed GR aware DTCO framework DTCO-pinOpt.

1. (*Conventional DTCO flow*) It uses the conventional placement and routing iteration flow for optimizing timing while trying to clean up DRVs as many as possible using a commercial physical design tool. (We used Cadence Innovus with the standard cells in redesigned ASAP 7nm library [7].)
2. (DTCO-pinOpt *using Step I only*) It uses our DTCO-pinOpt which applies Step I (i.e., pin pattern position optimization) only at the post placement stage.
3. (DTCO-pinOpt *using Steps I and II*) It uses our DTCO-pinOpt which exploits both of Steps I and II (i.e., pin pattern position and extension optimization) at the post placement and post routing stages.

Table I shows the comparison of the chip implementation produced by *Conventional DTCO flow*, DTCO-pinOpt *using Step I only*, and DTCO-pinOpt *using Steps I and II*, in which $\#Cell$ is the number of cell instances in circuits, $\#DRV$ is the number of pin access failures in terms of DRV count, *WL* is total wirelength, *PWR* is total power consumption, and *WTS* is the worst timing slack. The table shows that in comparison with *Conventional DTCO flow*, our DTCO-pinOpt is able to deliver chip implementations which have on average 5.88× fewer routing failures at no additional, wirelength, timing and

TABLE I
COMPARISON OF THE NUMBER OF PIN ACCESS FAILURES IN TERMS OF DRV COUNT, TOTAL WIRELENGTH (WL), TOTAL POWER CONSUMPTION (PWR), AND THE WORST TIMING SLACK (WTS) OF THE CHIP IMPLEMENTATIONS PRODUCED BY THE CONVENTIONAL GR-UNAWARE DTCO FLOW, OUR DTCO-pinOpt USING STEP I ONLY, AND OUR DTCO-pinOpt USING STEPS I AND II.

Circuit	#Cell	Conventional DTCO Flow				**DTCO-pinOpt** using Step I only				**DTCO-pinOpt** using Steps I and II			
		#DRV	WL	PWR	WTS	#DRV	WL	PWR	WTS	#DRV	WL	PWR	WTS
WB_DMA	3267	204	32947	0.81	0.03	2	32764	0.81	0.03	0	32761	0.81	0.03
AC97_TOP	10270	527	107716	3.84	0.03	2	106874	3.84	0.03	1	106869	3.84	0.03
WB_CONMAX	32014	2389	476958	1.86	0.03	542	479068	1.86	0.14	48	63233	1.56	0.53
LDPC	45470	1545	1625837	3.19	0.03	823	1627830	3.19	0.04	449	1628152	3.19	0.04
ETH_TOP	51919	3470	820216	5.21	0.10	1882	829744	5.22	0.07	1489	829459	5.22	0.05
AES_128	117900	5083	1595561	10.02	0.02	2179	1604221	10.04	0.02	1579	1604615	10.04	0.02
ECG	136942	9363	1539374	0.88	4.44	51	1539699	0.88	4.57	20	1539603	0.88	4.54
JPEG	579218	10621	3293629	3.52	-0.02	5911	3303031	3.53	0.07	4804	3302169	3.53	0.03
TATE_PAIRING	308080	18539	2960720	2.07	-1.61	184	2960933	2.07	-2.12	74	2960810	2.07	-2.12
Ratio (avg.)	-	1	1	1	met	**0.26**	1	1	met	**0.17**	0.90	0.98	met

power cost. Table II shows statistics on the number of cell instances in circuits which have been replaced with new cells produced by our DTCO-pinOpt at the placement stage to make M2 on-track alignment.

TABLE II
THE NUMBER OF CELL INSTANCES REPLACED WITH NEW CELLS PRODUCED BY DTCO-pinOpt AT THE PLACEMENT STAGE TO MAKE M2 ON-TRACK ALIGNMENT.

Circuit	DTCO-pinOpt	
	#Cell	#Replaced
WB_DMA	3267	1631 (50.08%)
AC97_TOP	10270	5060 (50.73%)
WB_CONMAX	32014	16012 (49.98%)
LDPC	45470	22711 (50.05%)
ETH_TOP	51919	25840 (50.23%)
AES_128	117900	59022 (49.94%)
ECG	136942	68603 (49.90%)
JPEG	279218	139405 (75.93%)
TATE_PAIRING	308080	154099 (49.98%)

IV. CONCLUSION

This work addressed a new important problem of standard cell optimization to improve pin accessibility for implementing a target chip that used a non-1:1 GR between gate poly pitch in cells and metal pitch of the first vertical metal layer to be used for routing. Precisely, contrary to the conventional approaches, in which they applied a local cell shifting to align routing tracks or used cells with longer pin patterns than necessary with no consideration of standard cell pin pattern optimization, we proposed a non-1:1 GR aware DTCO framework to resynthesize pin patterns on cells that were best fitted to the implementation of target design. In the meantime, it was shown through experiments that our proposed pin pattern optimization for improving pin accessibility under non-1:1 GR was very promising, considerably improving chip routability. As a future work, we want to develop a design methodology that is able to explore and find an optimal GR for a target circuit.

ACKNOWLEDGEMENT

This work was supported in part by Samsung Electronics Company, Ltd. under IO201216-08205-01, IO230102-04421-01, and IO221227-04376-01, in part by the National Research Foundation of Korea (NRF) grant funded by the Korea Government (MEST) under 2021-R1A2C2008864, in part by the Institute of Information and communications Technology Planning and Evaluation (IITP) grant funded by Korea government (MSIT) under 2021-0-00754, Software Systems for AI Semiconductor Design), and in part by the BK21 Four Program of the Education and Research Program for Future ICT Pioneers, Seoul National University. The EDA tool was supported by the IC Design Education.

REFERENCES

[1] J. D. O., "Cell architecture based on multi-gate vertical field effect transistor," in *Patent No 10,629,682, 2020.*, 2020.
[2] V. Vinay, M. Vangala, and L. T. Clark., "Asap7 predictive design kit development and cell design technology co-optimization," in *IEEE/ACM International Conference on Computer-Aided Design*, 2017.
[3] S. Jaewoo and Y. Shin, "Routability enhancement through unidirectional standard cells with floating metal-2," in *Design-Process-Technology Co-optimization for Manufacturability XI*, 2017.
[4] B. Yongchan, Z. Xuelian, P. Jan, and Z. Jia, "Improving performance, power, and area by optimizing gear ratio of gate-metal pitches in sub-10nm node cmos designs," in *IEEE Symposium on VLSI Tech.*, 2018.
[5] K. A. B, K. Jian, L. Wen-Hao, and X. Bangqi, "In-route pin access-driven placement refinement for improved detailed routing convergence," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 2021.
[6] X. Ruilong, P. Chanro, C. Richard, R. Robert, Z. Huimei, S. Iqbal, C. Adra, F. S. S. Chen, R. Kevin, B. Michael *et al.*, "Self-aligned gate contact (sagc) for cmos technology scaling beyond 7nm," in *IEEE Symposium on VLSI Tech.*, 2019.
[7] *Innovus User Guide*, 2016, https://www.cadence.com.
[8] T. C. Yu, S. Y. Fang, H. S. Chiu, K. S. Hu, P. H. Y. Tai, C. C. F. Shen, and H. Sheng, "Pin accessibility prediction and optimization with deep-learning-based pin pattern recognition," in *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 2020.
[9] *Calibre verification user's manual*, Graphics, Mentor, 2008.
[10] *Siliconsmart User Guide*, 2017, https://www.synopsys.com/implementation-and-signoff/signoff/siliconsmart.html.
[11] *Open-Source IP-Cores, www.opencores.org*, OpenCores, 2005.