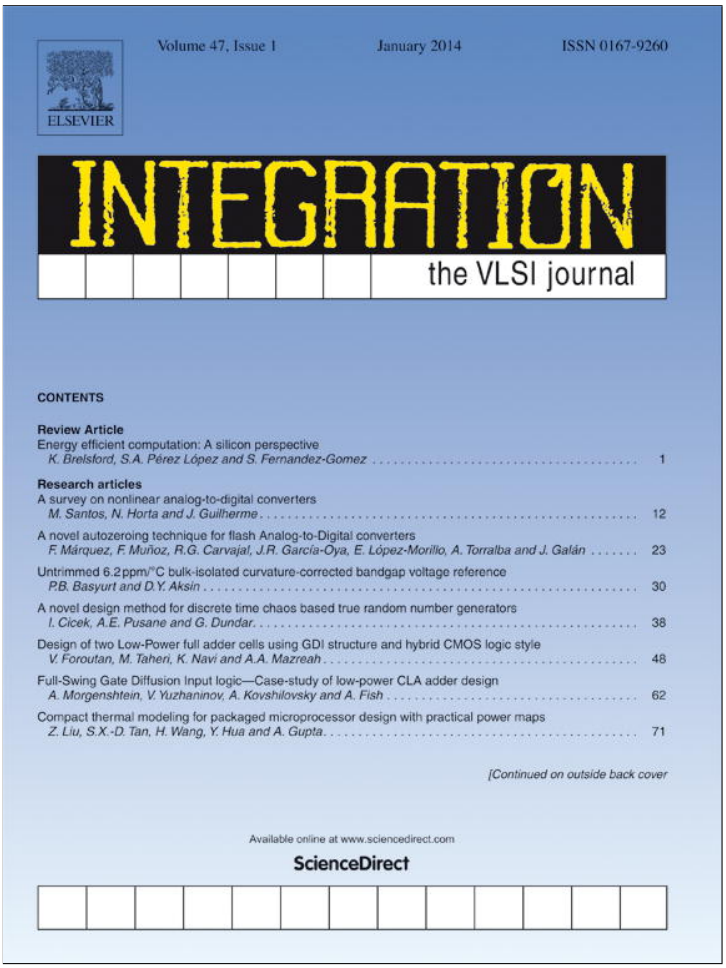
Provided for non-commercial research and education use.
Not for reproduction, distribution or commercial use.



Volume 47, Issue 1 January 2014 ISSN 0167-9260

ELSEVIER

INTEGRATION
the VLSI journal

CONTENTS

[Continued on outside back cover

Available online at www.sciencedirect.com
ScienceDirect

This article appeared in a journal published by Elsevier. The attached copy is furnished to the author for internal non-commercial research and education use, including for instruction at the authors institution and sharing with colleagues.

Other uses, including reproduction and distribution, or selling or licensing copies, or posting to personal, institutional or third party websites are prohibited.

In most cases authors are permitted to post their version of the article (e.g. in Word or Tex form) to their personal website or institutional repository. Authors requiring further information regarding Elsevier's archiving and manuscript policies are encouraged to visit:

http://www.elsevier.com/authorsrights

Author's personal copy

INTEGRATION, the VLSI journal 47 (2014) 115–122

Contents lists available at ScienceDirect

INTEGRATION, the VLSI journal

journal homepage: www.elsevier.com/locate/vlsi

Planar CMOS to multi-gate layout conversion for maximal fin utilization

Shmuel Wimer [a,b,*]

[a] Technion—Israel Institute of Technology, EE Department, Haifa, Israel
[b] Bar-Ilan University, Engineering Faculty, Rama-Gan, Israel

ARTICLE INFO

Article history:
Received 1 October 2012
Received in revised form
27 February 2013
Accepted 6 March 2013
Available online 21 March 2013

Keywords:
Multi-gate transistors layout
TriGate
FinFET
Hard-IP reuse

ABSTRACT

Multi-gate transistors enable the pace of Moore's Law for another decade. In its 22 nm technology node Intel switched to multi-gate transistors called TriGate, whereas IBM, TSMC, Samsung and others will do so in their 20 nm and 14 nm nodes with multi-gate transistors called FinFET. Several recent publications studied the drawing of multi-gate transistors layout. Designing new VLSI cell libraries and blocks requires massive re-drawing of layout. Hard-IP reuse is an alternative method taking advantage of existing source layout by automatically mapping it into new target technology, which was used in Intel's Tick-Tock marketing strategy for several product generations. This paper presents a cell-level hard-IP reuse algorithm, converting planar transistors to multi-gate ones. We show an automatic, robust transformation of bulk diffusion polygons into fins, while addressing the key requirements of cell libraries, as maximizing performance and interface compatibility across a variety of driving strength. We present a layout conversion flow comprising time-efficient geometric manipulations and discrete optimization algorithms, while generating manually drawn layout quality. Those can easily be used in composing larger functional blocks.

© 2013 Elsevier B.V. All rights reserved.

1. Introduction and motivation

Intel has lately delivered to market its Ivy Bridge microprocessor which includes new 22 nm multi-gate transistors, called *TriGate* [1]. Those are fundamentally different than the traditional planar bulk transistors used since the early days of MOS technology. TriGate enables the VLSI industry to continue the pace of Moore's Law for 14 nm, 10 nm and smaller feature-size technologies. TriGate transistor uses three gates wrapped around the silicon channel in a 3-D structure, delivering significant performance and energy efficiency improvement over planar transistors, while the transistors density remains intact. We subsequently use the terms transistor and device interchangeably. Fig. 1(a) shows the structure of a traditional planar transistor, while Fig. 1(b) shows the structure of a TriGate transistor comprising a single fin. Fig. 1(c) shows that TriGate transistors can have multiple fins connected together to increase total drive strength [1].

Other companies like IBM [2], TSMC [3], GlobalFoundries [4], Samsung [5] and STMicroelectronics [6] are also reporting on their progress in 3-D multi-gate transistors, known as *FinFET*. Those include a variety of technologies, some are similar to Tri-Gate, others comprise four terminal devices, and some are bulk while others are of Silicon on Insulator (SOI). All those technologies are expected to be used for products manufactured in 20 nm and 14 nm. We subsequently use the term multi-gate to mean both TriGate and FinFET.

While the physical structure of the various multi-gate transistor types are somewhat different of each other, they all share a common property, where the drain and sources of the underlying CMOS transistor are implemented by fins. From layout and mask design perspective both TriGate and FinFET look similar, where their fins must align to a uniform grid imposed on the entire chip to enable lithography. Fig. 2 is a photograph of an ensemble of TriGate transistors connected together.

The increasing interest in multi-gate transistors yielded lately few works related to their drawing. In [7] the drawing of a single transistor with multiple fins is described, while in [8,9] the layout drawing of complete cells is described. Drawing multi-gate devices must obey strict design rules enforcing the fins to align to a predefined grid common to the entire layout. In [10] the layout density obtained by using TriGate (3T) devices versus FinFET devices having four terminals (4T) was compared, concluding that 3T devices offer higher layout density. (3T devices are also known as connected-gate (CG) FinFET, while 4T are called isolated-gate (IG) FinFET.)

Former works dealt with the construction of multi-gate layouts from scratch, which is the common VLSI design scenario. There, cell library families comprising logic, sequential, memory, IO and other circuits are designed first and then used to build larger

* Corresponding author at: Technion—Israel Institute of Technology, EE Department, Haifa, Israel. Tel.: +972 4 8295744.
E-mail address: wimes@biu.ac.il

0167-9260/$ - see front matter © 2013 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.vlsi.2013.03.004

Author's personal copy

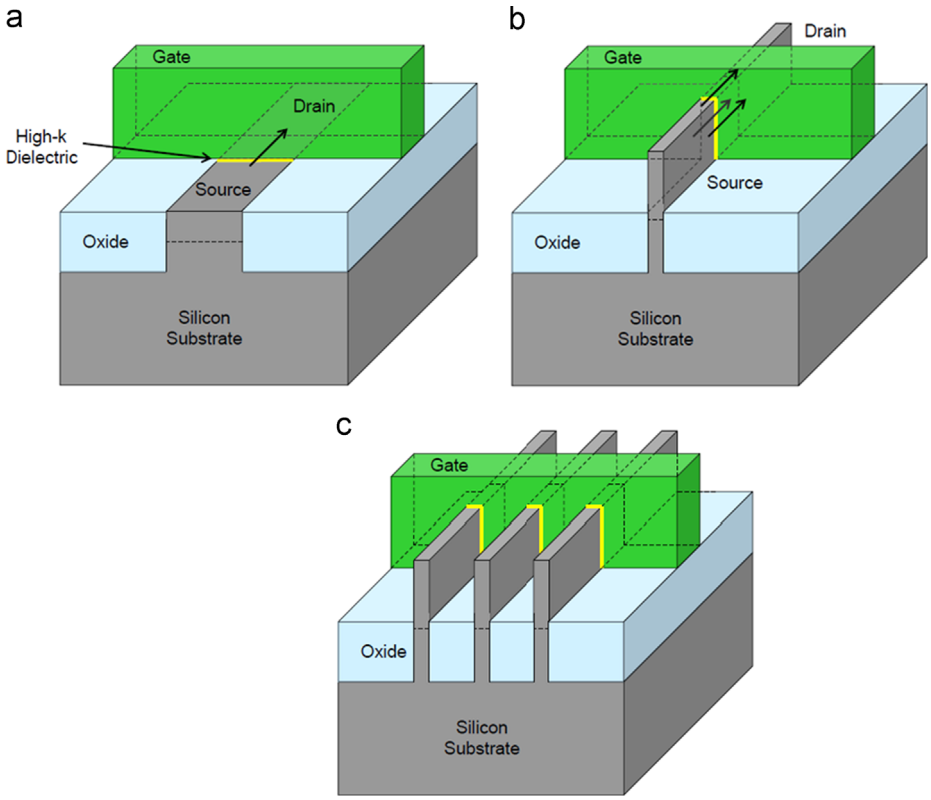


Fig. 1. Planar device in (a), single fin TriGate device in (b) and multiple fin device in (c) [1].

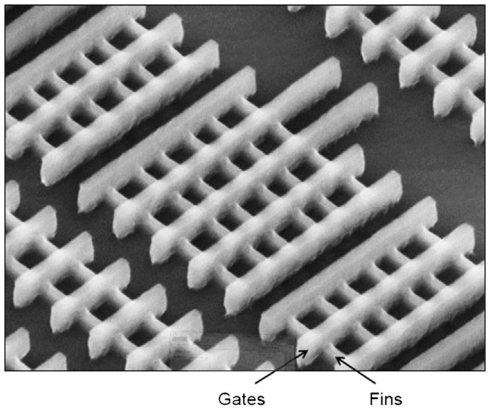


Fig. 2. TriGate transistors connected in a functional circuit [1].

functional blocks and systems. In a different scenario, known as *hard-IP reuse* [11,16], the physical layout of an existing chip, called source layout, is migrated into new target technology in an attempt to deliver the same functionality with improved energy efficiency and performance, but in lower cost. Such conversions were used at Intel as a part of its well-known Tick-Tock marketing strategy [11,12].

In a hard-IP reuse methodology called *cell-based*, the layout of the cells in the libraries are converted first and then are being placed and routed in an attempt to preserve their relative positions as in the source layout [13]. Converting the layout of cell libraries has the advantage of re-using the IP invested in their optimization, which mainly deals with the relative position of the transistors within the cells, their relative sizes and cell's interface. Such optimization is primarily affected by cell functionality and its underlying connectivity, while the specific technology in use has secondary impact on the optimization. Since cell's functionality and connectivity do not change across technology migration, layout topology is not suspected to significant changes. It is therefore beneficial to develop layout conversion algorithm supporting the transition from planar to multi-gate technology, and also from multi-gate to multi-gate for future technology migrations.

An early attempt for automatic conversion of planar CMOS SOI microprocessor into FinFET technology at 100 nm was reported in [14], but details were not provided. In [15] the authors describe an automatic process to replace the active diffusion regions of devices by appropriate fins to generate a legal and functional cell layout. Some oversizing of the active area takes place in order to legalize a fin in case it is not fully contained within the source active area. Though increasing the probability of legal and functional FinFET cell, it is a local transformation which does not take into account the entire cell devices simultaneously and their “competition” on the area. Furthermore, oversizing the active area as proposed in [15] still does not guarantee that all the required fins can indeed be materialized.

This work presents a planar to multi-gate layout conversion flow maximizing the fin utilization in the target cell. We shall address the main requirements from standard cell libraries while minimizing the amount of additional manual artwork required by a mask designer in case that conversion goals have not been met. The proposed flow is also applicable for migration of multi-gate source layout to multi-gate target layout across progression in process technologies, a situation expected for years to come. The rest of the paper is organized as follows. Next section presents the conversion problem and its tradeoffs. Section 3 outlines the conversion flow and briefly explains its various steps. Section 4 elaborates on the step of polygon rectangular decomposition, while Section 5 discusses the grid imposition algorithm and shows its optimality. Experimental results are discussed in Section 6.

Author's personal copy

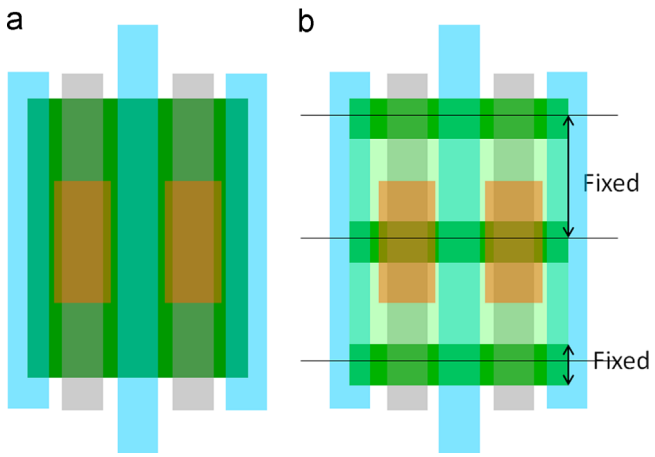


Fig. 3. Layout of a planar CMOS transistor and its multi-gate conversion.

2. The conversion of planar CMOS to multi-gate layout

A layout of a planar CMOS device is illustrated in Fig. 3(a). Its conversion into multi-gate, shown in Fig. 3(b), attempts to embed fins in the active area. The width of the fins (channel length) and their spacing are defined by the technology in hand and are common to the entire chip. The conversion must take this into account, such that plugging cells into blocks will automatically align the fins to that grid. The changes implied in the other layers are minor and will be mentioned when applicable.

The conversions of different planar transistors are not independent of each other. Fig. 4(a) shows illegal result occurred by converting each transistor alone, while Fig. 4(b) is a legal conversion obtained by considering transistors simultaneously. There is a problem however. While each planar transistor can accommodate three fins, only two are possible in a legal layout, thus degrading the driving strength of the circuit. More severely, when the source layout has very small devices, typically the keeper devices of sequential cells or those in min-delay buffers, it is not guaranteed that the resulting target layout will be feasible. This happens if there is no grid position (offset) over the source cell guaranteeing at least one fin per planar transistor. To minimize the chance of non-feasibility, the conversion flow is first oversizing the active area, expanding it as much as possible without violating design rules. This increases the chance for finding grid offset yielding feasible target layout.

Consider the example of a planar CMOS sequential cell illustrated in Fig. 5(a), where the gate and active layers are shown. In Fig. 5(b) the active areas have been vertically expanded together with the underlying gates. To preserve connectivity, diffusion contacts and first metal layer (not shown) are stretched accordingly. A uniform fin grid is then imposed on the cell and fins are populated as shown in Fig. 5(c). Fins resulting by a different grid offset are shown in Fig. 5(d), raising the question of which of the two offsets is better.

A clear difference between Fig. 5(c) and (d) is in their total number of fins, where in Fig. 5(c) there are 25 more fins than in Fig. 5(d). Having more fins is advantageous. First, after being converted the layout is extracted, simulated and further optimization may takes place by dropping some fins. Having initially more fins provides more degrees of optimization freedom, hence potentially obtaining a better cell. Secondly, libraries contain families of cells comprising same logic function with different driving strength. More fins in the output driver transistors ease the derivation of driving strength variants by simply dropping fins from the relevant transistors. The resulting cells still have the same footprint and interface, which is a big advantage, since it eases later timing fixes (called ECOs) by a simple cell replacement in layout.

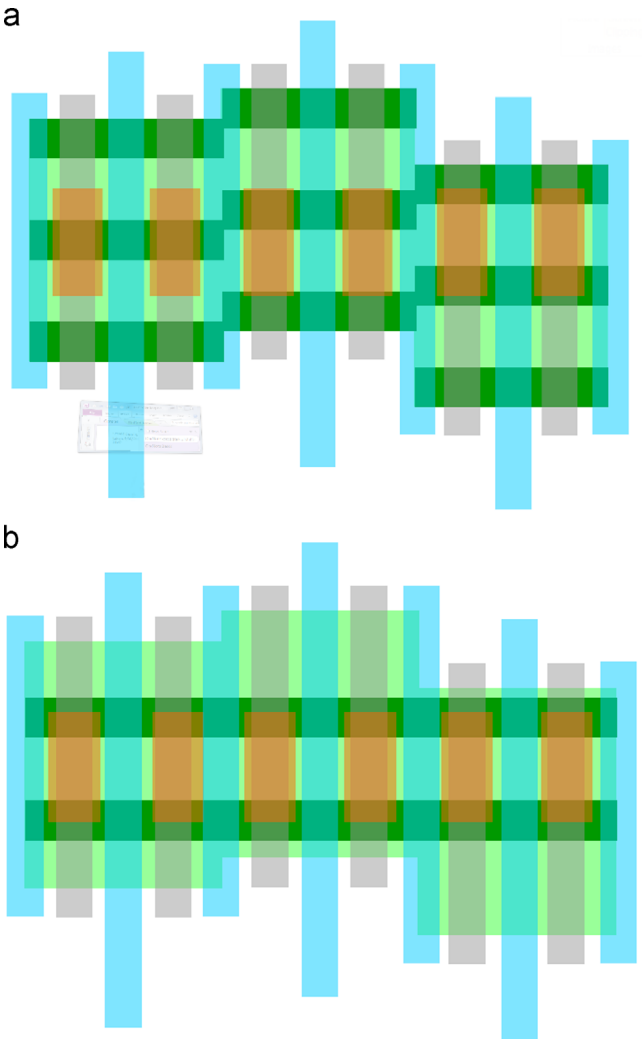


Fig. 4. Imposing a common grid on the entire layout. (a) Illegal - fins do not align and (b) legal - fins align.

Minimizing the number of transistors having less than a specified number M of fins is also an important objective. It can be supported by a simple modification of the algorithm maximizing the total number of fins. Setting $M = 1$ ensures that if there is a feasible solution where each transistor has one fin at least, it will be found by the algorithm. This is elaborated in Section 5.

3. Outline of conversion flow

The main steps of the conversion flow are outlined below. Their algorithmic details and optimality proofs are addressed in the next sections. We demonstrate the steps on the same layout used in Fig. 5. The conversion flow assumes that diffusion polygons (active layer) of the source layout are rectilinear, which is anyway the usual case.

3.1. Oversizing the active layer polygons

This step expands the n-type and p-type diffusion polygons in the vertical direction as much as possible, in an attempt to maximize the size of the planar transistors of the cell. Diffusion polygons are expanded together with their related gate polygons, diffusion contacts and straps of first metal layer. Such expansion

Author's personal copy

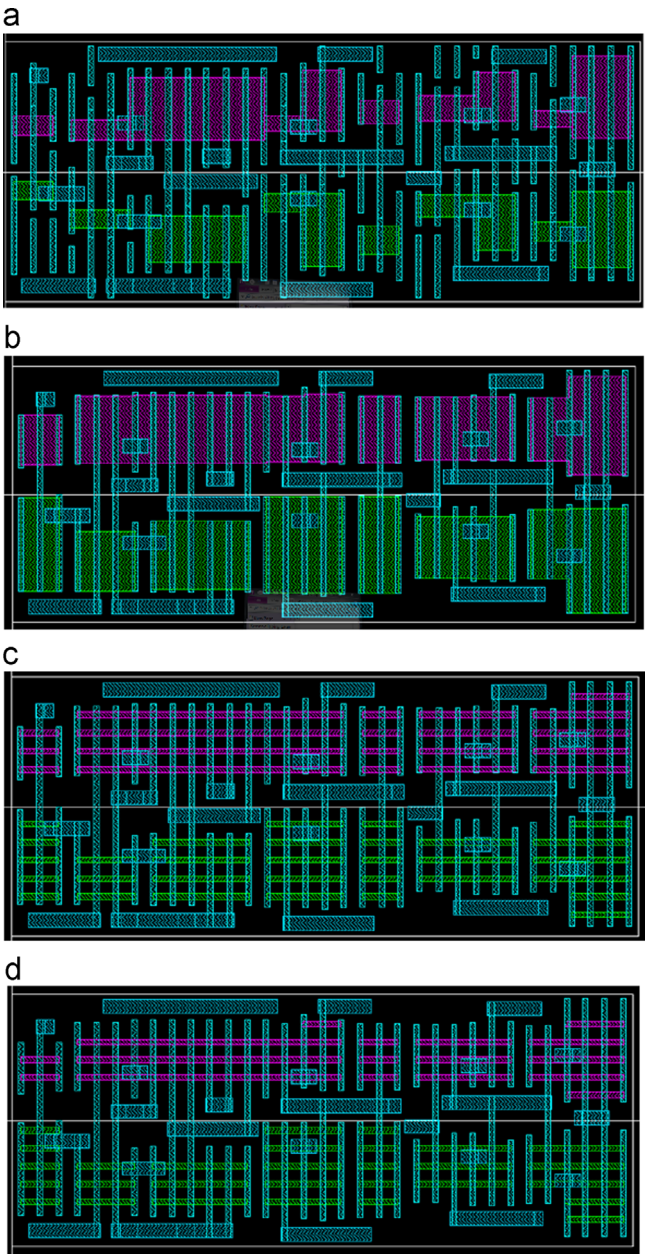


Fig. 5. Populating fins in active area. (For interpretation of the references to color in this figure, the reader is referred to the web version of this article.)

involves polygon oversizing and Boolean operations of the relevant layers, similar to those used for Design Rule Checking (DRC). This is performed with the aid of commercial EDA layout tools whose manuals specify the appropriate expressions to obtain the manipulations (e.g. [18,19]).

Diffusion expansion improves the chance of later feasible fin embedding and it provides other advantages mentioned in Section 2. The device expansion must preserve all the layout design rules and connectivity so the resulting layout is legal and functional. The step is illustrated by the transformation of the source layout in Fig. 5(a) to the expanded layout in Fig. 5(b).

Maximizing the size of transistors change their parameters in the source layout, but it is done only in an attempt to accommodate as many fins as possible in the target layout. The resulting fins are not necessarily all utilized. The exact setting of fin count per transistor is done by circuit simulation, combining parameters of multi-gate process technology and layout extraction. Appropriate design considerations are described in [23], Ch. 7.2.

3.2. Horizontal decomposition of diffusion polygons into maximal rectangles

This step simplifies the algorithm of fin maximization which follows. The decomposition of the diffusion expanded layout (see Fig. 5(b)) is shown in Fig. 6. The decomposition problem and its algorithmic solution are elaborated in Section 4.

3.3. Finding an optimal offset of fin grid

This step aims at maximizing the number of multi-gate fins embedded into the area occupied by the expanded planar transistors. Fig. 7 illustrates two grids imposed on the cell in different offsets. While the green grid implies the multi-gate layout shown in Fig. 5(c), the orange grid yields the layout in Fig. 5(d), which has 25 less fins, hence inferiors compared to the former. Section 5 describes a time-efficient algorithm that finds an optimal offset.

3.4. Replacing diffusion rectangles by fins

In this step the portions of the grid lines contained in the rectangles are sized to form legal fins with their width and spacing as determined by the process technology in hand. Since the grid lines are spaced in fin pitch (width+space), the resulting fins are guaranteed to adhere the design rules. The source and drain of the fins corresponding to the same transistor are automatically connected by first metal layer straps stretched in Section 3.1 (not shown).

3.5. Legalization of multi-gate layout

The former steps apply geometric manipulations preserving connectivity and accounting for fin and other design rules. Some fixes related to layout methodology, such as setting cell's origin to align to grid line, are still required. This ensures that once cells are placed together, all fins will perfectly abut and align to a global grid imposed on the entire layout. Design rule violations related to other than fin layers can still happen. Their fixing can be handled automatically by layout compaction EDA tools [16] or manually. This is not further elaborated as it involves ordinary design activities.

Once the multi-gate transistors have been legally converted, their remaining interconnections are completed. In the conversion

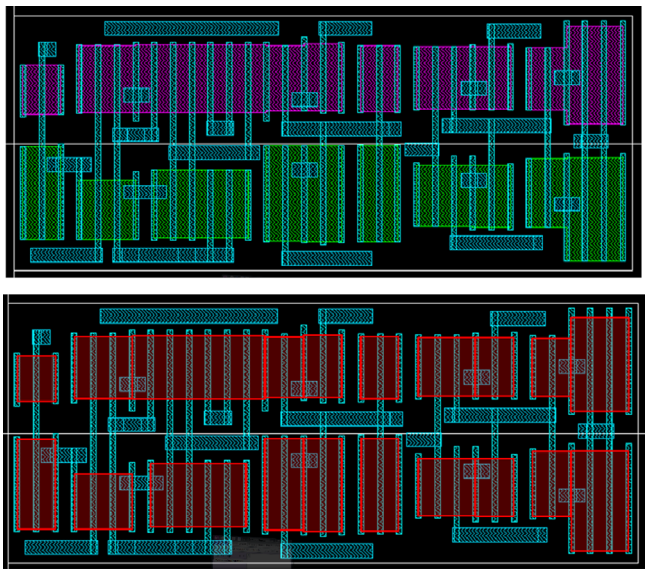


Fig. 6. Decomposing diffusion polygons into maximal vertical rectangles.

Author's personal copy

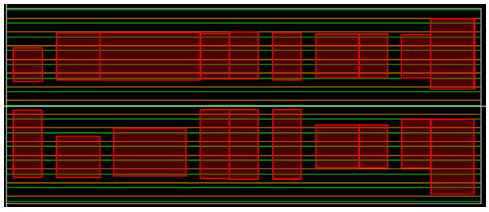


Fig. 7. Imposition of fin grid.

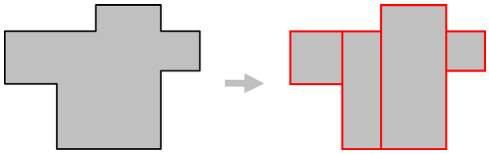


Fig. 8. Horizontal decomposition of a polygon into maximal rectangles.

of library cells this task requires small manual effort. In 22 nm and smaller process technologies, the low-level interconnecting layers are gridded and aligned to the horizontal fins and vertical gates. Gridding of critical layers significantly reduces the total physical design space available and makes restrictive design possible [24]. Automation of routing migration is also possible and available by EDA commercial tools [16].

The above flow describes planar CMOS transistors source layout conversion into multi-gate transistors target layout. There is no restriction however in using it for multi-gate to multi-gate conversion expected for next process technologies. This can simply be done by first enclosing the fins comprising individual transistors in source layout by smallest rectangles. The problem is known as finding the *external contour of a union of rectangles*. For N fins this is done in $\theta(N \log N)$ optimal time and $O(N \log N)$ worst-case space [[20], Ch. 8.7]. The resulting rectangles are then merged to obtain input layout which looks as in Fig. 5(a). The rest of the flow does not change.

4. Horizontal decomposition into maximal rectangles

We subsequently define the polygon decomposition problem mentioned in Section 3.2 and elaborate the algorithm solving it. Let E be the sets of the horizontal edges of a rectilinear polygon P, ordered lexicographically by their left end abscissa as a primary key and by their ordinate as a secondary key. We say that $\mathbf{R} = R_i$ is a *horizontal decomposition* of P into *maximal rectangles* if:

1. $\cup R_i = P$,
2. $R_i^\circ \cap R_j^\circ = \phi$, $i \neq j$, where R° is the interior of R.
3. The top and bottom edges of R_i are fully contained in edges of P.
4. Any rectangle R satisfying 3 such that $R_i \subset R \subset P$ implies $R = R_i$, namely, $\mathbf{R} = R_i$ are maximal. An example of such decomposition is shown in Fig. 8.

2D rectangular polygon decomposition is a common problem arising in VLSI layout and data processing for mask generation. A fast algorithm that solves the problem was presented in [21]. In its most general form the 2D problem is NP-hard and a comprehensive discussion of this topic can be found in [22]. In our context the decomposition turns into 1D problem, for which a time optimal algorithm is subsequently described.

For an edge $e \in E$ we denote by $x_l(e)$, $x_r(e)$ and $y(e)$ its left end abscissa, right end abscissa and ordinate, respectively. Let E be sorted in ascending order by the key $[x_l(e), y(e)]$ and let $\min(E)$ be the smallest element in E. Notice that E is in some way a description of P whose contour can be retrieved by a linear traversal of E (after sorting). To handle efficiently deletion, insertion and retrieval of smallest elements, E is implemented as a heap, so the time of operations on E will not exceed $O(\log|E|)$ [[17], Ch. 7].

The algorithm of decomposition into maximal rectangles is described by the pseudo code in Fig. 9, and its various steps are illustrated in Fig. 10. After initialization of $\mathbf{R}$ and sorting E in ascending order in line 1, the algorithm iterates from lines 2 to 8 where at each iteration a new maximal rectangle is found and subtracted from P. The leftover of P is updated accordingly. The maximal rectangles are shown in Fig. 10(a)–(c) for the first three iterations of the algorithm. The algorithm ends once P is consumed, or equivalently, when E is emptied. Notice that $P = \varnothing$ if and only if $E = \varnothing$.

The first two edges of E (having smallest left end abscissa), e_1 and e_2 respectively (shown in red in Fig. 10) are found in lines 3 and 4 and then deleted from E. The edges e_1 and e_2 define the bottom and top edges of the maximal rectangle. Notice that there is $x_l(e_1) = x_l(e_2)$. The left edge abscissa a maximal rectangle R is defined in line 5 and it is any of $x_l(e_1)$ or $x_l(e_2)$ (those are equal). Its right edge abscissa is the smaller among $x_r(e_1)$ and $x_r(e_2)$. Line 6 defines the ordinates of the bottom and top edges of$R$. The maximal rectangle R thus defined is then added to $\mathbf{R}$ in line 7.

If $x_r(e_1) = x_r(e_2)$ R has effectively been subtracted from P at lines 3 and 4 by the deletion of the bottom and the top edges e_1 and e_2 from E. This is the case in Fig. 10(c). If however $x_r(e_1) \neq x_r(e_2)$ a portion of the longer edges among e_1 and e_2 is inserted to E in lines 8 and 9 as illustrated in Fig. 10(a) and (b), where the edge inserted to E is shown in blue.

Considering the time complexity of the algorithm, let P have p vertices and hence $p/2$ vertical and $p/2$ horizontal edges. Sorting E in line 1 and storing it in a heap takes $O(p \log p)$ time. An iteration of the decomposition algorithm consumes one vertical edge, hence there are $p/2$ iterations. Finding the smallest element in a heap in lines 3 and 4 takes $O(1)$ time, while deletions in lines 3 and 4, and insertions in lines 8 and 9 take $O(p \log p)$, which altogether yields $O(p \log p)$ time complexity. The time complexity is optimal and its proof is similar to that found in [20], Ch. 8.5.

5. Finding optimal offset of fin grid

It was demonstrated in Figs. 7 and 5(c) and (d) that fin grid offset affects the number of implied fins, and the advantage of maximizing those has been discussed. We subsequently present an algorithm that finds that maximum.

Shown in Fig. 11, let δ be the center-to-center distance between fins, which defines the fin grid imposed on the rectangles obtained

1. $Initialization:\ \mathbf{R} = \varnothing;\ sort(E);$
2. $while\ (E \neq \varnothing)\ \{$
3. $\quad e_1 = \min(E);\ Delete(e_1, E);$
4. $\quad e_2 = \min(E);\ Delete(e_2, E);$
5. $\quad x_l = x_l(e_1);\ x_r = \min\{x_r(e_1), x_r(e_2)\};$
6. $\quad y_b = y(e_1);\ y_t = y(e_2);$
7. $\quad \mathbf{R} = \mathbf{R} \cup R : [x_l, x_r] \times [y_b, y_t];$
8. $\quad if\ (x_r(e_1) > x_r(e_2))\ \{ Insert(e : [x_r(e_2), x_r(e_1)] \times y(e_1), E) \}$
9. $\quad elseif\ (x_r(e_2) > x_r(e_1))\ \{ Insert(e : [x_r(e_1), x_r(e_2)] \times y(e_2), E) \}$
10. $\}$

Fig. 9. Algorithm of horizontal decomposition into maximal rectangles.

Author's personal copy

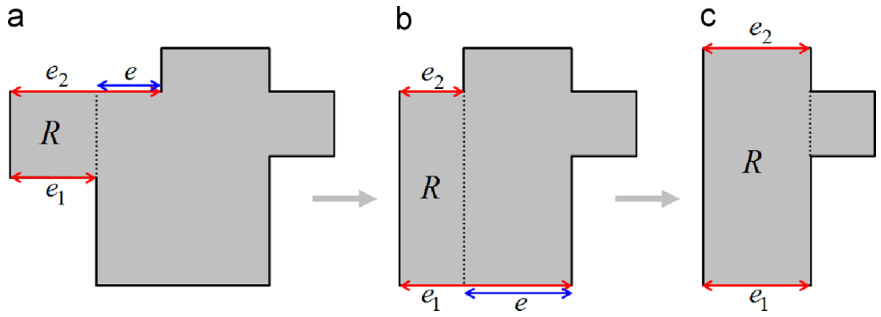


Fig. 10. Illustration of the first three iterations. (For interpretation of the references to color in this figure, the reader is referred to the web version of this article.)

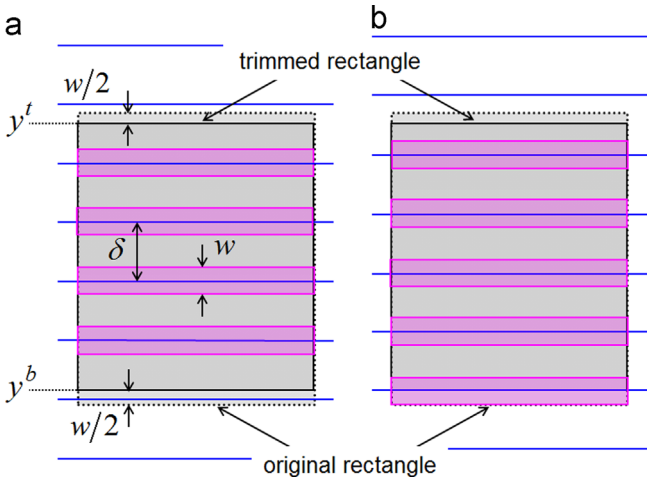


Fig. 11. Different offsets of fin grid yield different number of fins.

in Section 3.2 of the conversion flow described in Section 3. Let w be the fin size drawn in layout. The grid defines the legal location for fins. To ease the finding of the optimal grid offset, the rectangles are first trimmed in their bottom and top sides by $w/2$ as shown in Fig. 11, where the dotted border is the original rectangle and the solid border it the trimmed one. Legal fins are then defined by the intersection of the grid with the trimmed rectangles. We subsequently treat the rectangles as already being trimmed.

Let y^b and y^t be the ordinates of R's bottom and top edges as shown in Fig. 11. It is convenient to define the lower-left corner of the cell layout as an origin ($y=0$). The bottom and top ordinates of R are defined with respect to that origin. The expression

$$N=\left\lfloor\frac{y^t-y^b}{\delta}\right\rfloor \tag{1}$$

defines the number of grid lines intersecting with R which can be either N or $N+1$, depending on the offset of the grid with respect to R. Correspondingly, the number of legal fins embedded in R is TN or $T(N+1)$, where $T$is the number of transistors existed in R in the source layout. This is demonstrated in Fig. 11 where$N=4$. While the offset in Fig. 11(a) yields $4T$ fins, it yields $5T$ fins in Fig. 11(b). The grid offset ε can be any value in the range $0\le\varepsilon\le\delta$, and the number of fins in R is therefore a two-valued function

$$f(\varepsilon):[0,\delta]\to\{TN,T(N-1)\}. \tag{2}$$

Evidently, while some ε can be favored for a rectangle $R_i\in\mathbf{R}$, yielding $T_i(N_i+1)$ fins, it can be is inferior for another rectangle $R_j\in\mathbf{R}$, yielding only T_jN_j fins. Let the layout contain r maximal rectangles R_i, $1\le i\le r$, and let f_i be their corresponding fin count functions defined in (2). Finding the optimal grid offset is formulated as follows:

$$\text{maximize } F(\varepsilon)=\sum_{i=1}^{r} f_i(\varepsilon),\quad \text{subject to : } 0\le\varepsilon\le\delta. \tag{3}$$

It follows from (2) and (3) that $\sum_{i=1}^{r}T_iN_i\le F(\varepsilon)\le\sum_{i=1}^{r}T_i(N_i+1)$. The optimization problem in (3) is solved by transforming it into the following interval density problem. Notice that while the grid offset is changing continuously in the range $0\le\varepsilon\le\delta$, $f_i(\varepsilon)$ can change between T_iN_i and $T_i(N_i+1)$ at most twice. There exist ε'_i and ε''_i, satisfying $0\le\varepsilon'_i\le\varepsilon''_i\le\delta$ such that either

$$f_i(\varepsilon)=\begin{cases} T_iN_i & 0\le\varepsilon<\varepsilon'_i\\ T_i(N_i+1) & \varepsilon'_i\le\varepsilon<\varepsilon''_i\\ T_iN_i & \varepsilon''_i\le\varepsilon\le\delta\end{cases},\quad \text{or} \tag{4a}$$

$$f_i(\varepsilon)=\begin{cases} T_i(N_i+1) & 0\le\varepsilon<\varepsilon'_i\\ T_iN_i & \varepsilon'_i\le\varepsilon<\varepsilon''_i\\ T_i(N_i+1) & \varepsilon''_i\le\varepsilon\le\delta\end{cases}. \tag{4b}$$

whether (4a) or (4b) applies for a rectangle R_i depends on its position with respect to the grid origin. The situations in (4a) or (4b) are illustrated in Figs. 12(a) and (b), respectively. In (4a) the increase of ε from 0 to δ results in a grid line to first enter R_i from bottom as shown in Fig. 12(a). In (4b) the increase of ε from 0 to δ results in a grid line to first leave R_i from its top as shown in Fig. 12(b).

The value ε'_i defined in (5) below is the grid offset causing a grid line to enter R_i in (4a), thus adding T_i fins, and to leave R_i in (4b), thus subtracting T_i fins,

$$\varepsilon'_i=\min\left\{y_i^b-\delta\left\lfloor\frac{y_i^b}{\delta}\right\rfloor,y_i^t-\delta\left\lfloor\frac{y_i^t}{\delta}\right\rfloor\right\}. \tag{5}$$

The value ε''_i defined in the following equation is the grid offset causing the grid line to leave R_i in (4a), thus subtracting T_i fins, and to enter R_i in (4b), thus adding T_i fins,

$$\varepsilon''_i=\max\left\{y_i^b-\delta\left\lfloor\frac{y_i^b}{\delta}\right\rfloor,y_i^t-\delta\left\lfloor\frac{y_i^t}{\delta}\right\rfloor\right\}. \tag{6}$$

It follows from (4a) and (4b) that $F(\varepsilon)$ defined in (3) can change its values only at $\{0,\varepsilon'_1,\varepsilon''_1,\ldots,\varepsilon'_r,\varepsilon''_r\}$. Notice that $F(\delta)=F(0)$ since the grid offset is periodic. Let us sort $\{0,\varepsilon'_1,\varepsilon''_1,\ldots,\varepsilon'_r,\varepsilon''_r\}$ in ascending orders and denote the sorted values by $\varepsilon_0=0\le\varepsilon_1\le\varepsilon_2\le\ldots\le\varepsilon_{2r-1}\le\varepsilon_{2r}$. The function $F(\varepsilon)$ can change its value $2r+1$ times at most. To find the optimal ε where it is maximized there is no need to compute $F(\varepsilon)$, but rather cumulate its increments or decrements at ε_i, $0\le i\le 2r$. While an offset defined by (5) is increasing $F(\varepsilon)$ by T_i, an offset defined by (6) is decreasing it by T_i. By a linear traversal over the sorted list $(\varepsilon_0,\varepsilon_1,\varepsilon_2,\ldots,\varepsilon_{2r-1},\varepsilon_{2r})$ the optimal value ε^* maximizing the total number of fin is found. To consider the time complexity of finding the optimal offset,

Author's personal copy

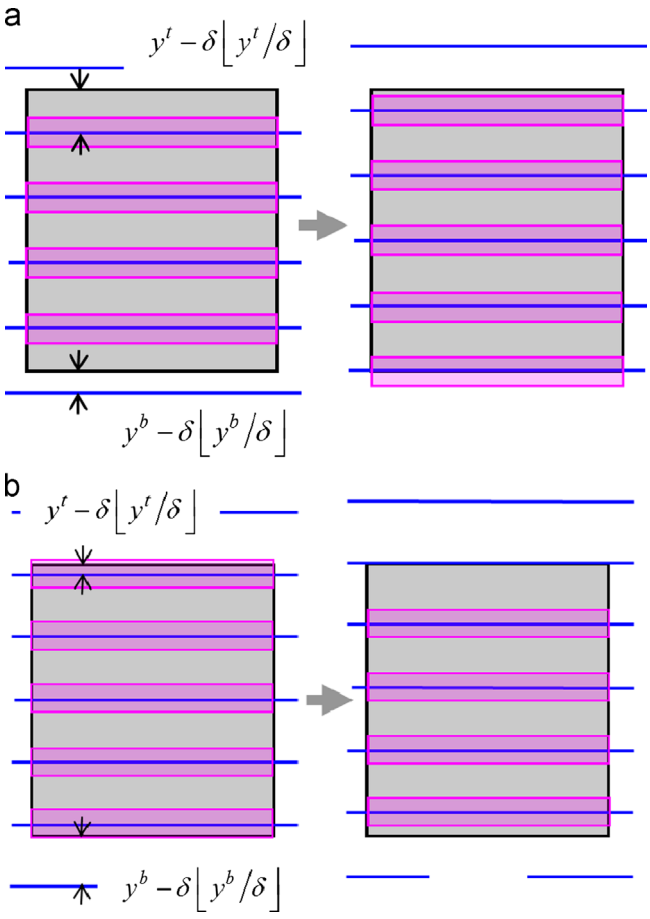


Fig. 12. Grid offsets adding fins and subtracting fins, depending on relative position of a rectangle with respect to grid origin.

Table 1
Grid offsets yielding minimum and maximum fin count.

Circuit type	Devices in schematics	Devices in layout	Min fins	Max fin	(Max–Min)/ Min (%)
Full Adder	32	40	157	202	29
Manchester Carry Chain	19	23	91	117	28
D-FlipFlop	26	32	96	125	30
D-FlipFlop (Async. Reset)	30	38	120	149	24
D-FlipFlop (Scan-Chain)	46	56	172	219	27

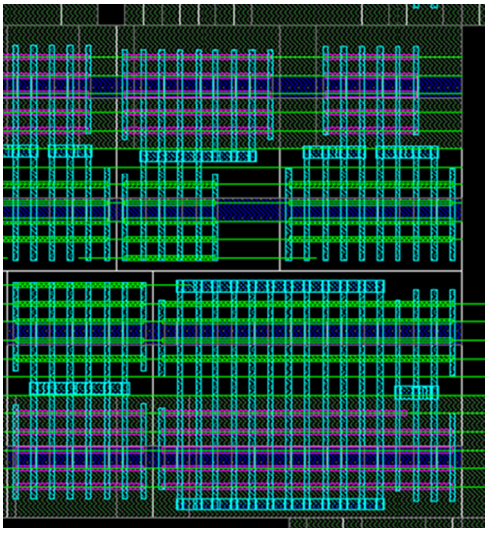


Fig. 13. Multi-gate cells placed in a larger block.

notice that while the computation of ε^* takes $O(r)$ time, the presorting of the offsets $\{0, \varepsilon'_1, \varepsilon''_1, \ldots, \varepsilon'_r, \varepsilon''_r\}$ takes $O(r \log r)$ time.

The above algorithm can easily be modified to minimize the number of transistors having less than a specified number M of fins. Let us define $\mathbf{R}^M = \{R_i | N_i = M-1\}$. It follows from (4a) and (4b) that only rectangles $R \in \mathbf{R}^M$ should be considered by the algorithm, since a rectangle $R \notin \mathbf{R}^M$ will either have at most $M-1$ or at least M fins, regardless of the grid offset. By restricting the problem to $\mathbf{R}^M$, the optimal offset can be found by maximizing $F(\varepsilon)$ as before. The special case where $M = 1$ guarantees that if there is a feasible solution where each transistor has one fin at least, the algorithm will find it.

6. Results

The conversion flow has been implemented and tested on many cell families, including sequential ones containing small keepers. In all cases we tried, the flow generated feasible layout. In most cases the grid offset achieved the primary goal of maximizing the total number of fins, yielding a feasible layout where each transistor has one fin at least. For those cases where maximum fins resulted illegal layout (usually a case of a small keeper in a sequential cell) we used the modification described in Section 5, which could always resolve the problem.

Table 1 shows the range of fin count that can be obtained by grid offsets. Five complex cells have been experimented, for each the count of schematics and layout transistors is shown. The reason for the difference between schematics and layout is due to the output transistors comprising several parallel connected legs, counted in the layout as distinct transistors. We observed a consistent increase of the fin count of 25–30% between grid offsets maximizing and minimizing the total number of fins. Notice that the same algorithm that finds the offset maximizing the number of fins, is finding as a by-product the offset minimizing their number. Due to the low polynomial complexity of the algorithms involved in the flow and the small size of library cells (hundreds to thousands polygons), run-time is negligible.

Fig. 13 shows a layout clip taken from a larger block in which the cells produced by the migration flow have been placed in their original position in the source layout, as a part of a cell-based layout migration flow. Notice that the underlying cells nicely fit each other, thus ensuring that all the fins abut and are placed on a grid common to the entire target layout.

Acknowledgment

The author is grateful for the useful comments made by the anonymous referees, which helped in improving the manuscript.

References

[1] Intel's Revolutionary 22 nm Transistor Technology, ⟨http://download.intel.com/newsroom/kits/22 nm/pdfs/22nm-Details_Presentation.pdf⟩, May 2011.
[2] Fins on Transistors Change Processor Power and Performance, ⟨http://ibmresearchnews.blogspot.co.il/2012/07/fins-on-transistors-change-processor.html⟩, July 2012.
[3] C.-Y. Chang, et al., A 25-nm gate-length FinFET transistor module for 32 nm node, in: Proceedings of IEEE International Electron Device Meeting, December 2009, pp. 293–296.
[4] GlobalFoundries intros 14 nm Process with FinFET Transistors, ⟨http://www.digitimes.com/news/a20120921PR200.html⟩, September 2012.

Author's personal copy

[5] Samsung, IBM and GlobalFoundries Look to the Future: A Report from the Common Platform Technology Forum, ⟨http://www.electroiq.com/articles/sst/2012/03/samsung-ibm-and-globalfoundries-look-to-the-future-a-report-from-the-common-platform-forum.html⟩, March 2012.
[6] T. Poiroux, et al., Multigate silicon MOSFETs for 45 nm node and beyond, Solid-State Electronics 50 (2006) 18–23.
[7] D.M. Fried, W.C. Leipold, E.J. Nowak, FinFET Layout Generation, US Patent 6,662,350, December 2003.
[8] S.T. Becker, Cell Circuit and Layout with Linear FinFET Structure, US Patent App. 12/775,429, November 2010.
[9] W. Xiong, et al., Integrated Circuit with Aligned NMOS and PMOS FinFET Sidewall Channel, US Patent App. 13/425,082, July 2012.
[10] M. Alioto, Comparative evaluation of layout density in 3 T, 4 T, and MT FinFET standard cells, IEEE Transactions on VLSI Systems 19 (5) (2011) 751–762.
[11] R. Nitzan, S. Wimer, AMPS and SiClone integration for implementing 0.18um to 0.13um design migration, in: Proceedings of the Synoposys Users Group (SNUG) Conference, San Jose CA, March 2002.
[12] Intel's Tick-Tock, 2011, Model, ⟨http://www.intel.com/technology/tick-tock/index.htm⟩.
[13] J. Zhu, F. Fang, Q. Tang, Calligrapher: a new layout-migration engine for hard intellectual property libraries, IEEE Transactions on CAD of Integrated Circuits and Systems 24 (5) (2005) 1347–1361.
[14] T. Ludwig, et al., FinFET technology for future microprocessors, in: Proceedings of IEEE International SOI Conference, September 2003, pp. 33–34.
[15] J.-J. Shen, et al., Automatic Layout Conversion for FinFET Device, US Patent App. 12/780,060, November 2010.
[16] Lay out Migration and DRC Correction, 2012 ⟨http://www.sagantec.com/nmigrate%20datasheet%20v2.pdf⟩.
[17] T.H. Cormen, C.E. Leiserson, R.L. Rivest, Introduction to Algorithms, MIT Press, US55 Hayward Street Cambridge, MA 02142-1315 USA, 1990.
[18] Calibre Verification User's Manual, Mentor Graphics, 2008, Mentor Graphics Corporation 8005 S.W. Boeckman Road, Wilsonville, Oregon 97070-7777.
[19] Assura Physical Verification User Guide, Cadence Design Systems, 2665 Seely Ave, San Jose, CA 95134, United States Cadence, 2011.
[20] F.P. Preparata, M.I. Shamos, Computational geometry: an introduction Springer-Verlag New York, Inc. New York, NY, USA ©1985ISBN:0-387-96131-3, Springer, 1985.
[21] S. Nahar, S. Sahni, Fast algorithm for polygon decomposition, IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems 7 (4) (1988) 473–483.
[22] M.J. Keil, Polygon decomposition, Handbook of computational geometry North-Holland Publishing Co. Amsterdam, The Netherlands, The Netherlands ©2000,vol. 2, 2000.
[23] G. Knoblinger, M. Fulde, C. Pacha, Multi-gate MOSFET circuit design, in: MLA; Colinge, J-P., ed. FinFETs and other multi-gate transistors. Springer, 2007. APA; Colinge, J. P. (Ed.). (2007). FinFETs and other multi-gate transistors. Springer. Chicago Colinge, J-P., ed. FinFETs and other multi-gate transistors. Springer, 2007. ,FinFETsandOtherMulti-GateTransistors,Springer, 2008.
[24] D. Abercrombie, P. Elakkumanan, L. Liebmann, Restrictive Design Rules and Their Impact on 22 nm Design and Physical Verification, Electronic Design Processes Workshop, April 9th–10th 2009.

Shmuel Wimer received the B.Sc. and M.Sc. degrees in mathematics from Tel-Aviv University, Tel-Aviv, Israel, and the D.Sc. degree in electrical engineering from the Technion-Israel Institute of Technology, Haifa, Israel, in 1978, 1981 and 1988, respectively. He worked for 32 years at industry in R&D, engineering and managerial positions, for Intel (1999-2009), Sagantec (1997-1999), microCAD (1994-1997), IBM (1985-1994), National Semiconductor(1981-1985) and Israeli Aircraft Industry (IAI) (1978-1981). He is presently an Associate Professor with the Engineering Faculty, Bar-Ilan University, Ramat-Gan, Israel, and an Associate Visiting Professor with the Electrical Engineering Faculty, Technion. His areas of interests include VLSI circuits and systems design optimizationand combinatorial optimization.