3D IC CAD Placement Flows and Algorithms Yielding Improved PPA

Nikolaos Sketopoulos

Department of Electrical and Computer Engineering
University of Thessaly

A thesis submitted in fulfillment of the requirements for the degree of
Doctor of Philosophy

September 2021

Institutional Repository - Library & Information Centre - University of Thessaly
30/12/2021 10:19:35 EET - 137.108.70.14

Institutional Repository - Library & Information Centre - University of Thessaly
30/12/2021 10:19:35 EET - 137.108.70.14

I would like to dedicate this thesis to my mother . . .

Institutional Repository - Library & Information Centre - University of Thessaly
30/12/2021 10:19:35 EET - 137.108.70.14

Institutional Repository - Library & Information Centre - University of Thessaly
30/12/2021 10:19:35 EET - 137.108.70.14

Acknowledgements

I would like to thank my mentor, Prof. Christos Sotiriou, whose encouragement, supervision, and support from the preliminary to the concluding level enabled me to pursue this research. I would like to express my gratitude to all of the members of my dissertation committee for their contributions at various phases of my academic career. Prof. Vasilis Pavlidis, in particular, deserves special thanks for his insightful comments on this study.

Furthermore, I wish to thank the members of the Circuits and Systems Lab, of the University of Thessaly, for the warm and productive collaboration. My special thanks go to George Rafael Goundroumanis, an undergraduate student who provided the METIS tool results. Moreover, I would like to thank the "ΔΕΚΑ" organization, of the University of Thessaly, for the one-year financial support. I would also like to thank Prof. Emre Salman from Stony Brook University for his valuable discussions and for providing the 3D nanoCAS library.

Finally, I would like to express my gratitude to my family, friends, and Maria for their never-ending love and support.

Institutional Repository - Library & Information Centre - University of Thessaly
30/12/2021 10:19:35 EET - 137.108.70.14

Institutional Repository - Library & Information Centre - University of Thessaly
30/12/2021 10:19:35 EET - 137.108.70.14

Abstract

In recent years, the needs for more processing power, faster circuits, and less power consumption lead the semiconductor industry to investigate innovating methodologies and technologies to continue the chip scaling process.

Moore's law, indicating that every 18 months the number of transistors is duplicated [1], comes to an end. The size of the transistors will not be able to be further reduced, while the manufacturing cost will continue to increase for each new technology node. In parallel, wires' delay cannot be scaled as transistors can, making the adoption of new technology nodes more challenging.

A solution allowing higher density is to stack transistors in multiple tiers, by utilizing a third dimension. Three-Dimensional (3D) integrated circuits (ICs) allow chip scaling, just like skyscrapers can accommodate more and more people at the same building block. By designing 3D ICs, there is neither need for new material usage nor is it necessary to spend large amounts of money on investigating smaller technological nodes. So, mature technology nodes can be used for 3D ICs, as the benefits of stacking transistors in multiple tiers may surpass the total benefits of utilizing new technology nodes.

Although the 3D ICs design has been around for decades, it is not being employed in mass manufacturing. Therefore, the semiconductor industry will have to deal with a number of new challenges due to the introduction of the third dimension. One of these challenges is higher density which results in higher circuit temperature, while the efficient linking of different tiers is another challenge. The absence of Electronic Design Automation (EDA) tools is another impediment to the mass manufacture of 3D ICs. Most of the attempts to design a 3D IC used traditional two-dimensional EDA tools and physical design flows, with modifications. However, circuit quality is low when the convectional two-dimensional flows and tools are used, as they do not inherently take the third dimension into consideration.

Our objective in this study is to investigate and evaluate different methodologies that provide the ability to separate a design into tiers, *i.e.* the tier assignment problem. This is one of the first steps for the 3D integration, while each methodology significantly impacts integration quality. Our main target has been to improve power consumption, circuit performance, circuit size, and achieve a limited number of interconnections across several tiers.

Institutional Repository - Library & Information Centre - University of Thessaly
30/12/2021 10:19:35 EET - 137.108.70.14

Firstly, we investigated several Block-Based techniques to divide and tackle the problem of assigning a large circuit to tiers. Since modern designs contain billions of standard cells, designing a chip that takes all of the components at once into account is quite challenging. To simplify the three-dimensional design, transitional two-dimensional divide-and-conquer approaches are employed. So, we developed and consequently we analyze three-dimensional floorplanning and clustering algorithms for tier assignment in this study, to handle very large designs. This floorplanning algorithm aims to partition RTL functional modules into tiers of equal size, while the clustering algorithm aims to generate well-formed (*e.g.* balanced, low connectivity) groups of standard cells that will be allocated to tiers.

Despite the fact that Block-Based techniques can handle very large designs, their quality is restricted due to their higher level of circuit abstraction. So, we also investigated Fine-Grained techniques as partitioning and three-dimensional legalization. Fine-grained approaches provide greater freedom, which can lead to better solutions. However, these approaches must be able to handle today's very large circuits, that contain billions of standard cells. But, the Fine-Grained techniques that we propose to assign standard cells to tiers do handle these very large designs.

Finally, we evaluate the merits and limitations of suggested techniques, algorithms, and flows by comparing the placed and routed designs. We also provide assistance on how to apply the methods, algorithms, and flows on three-dimensional circuits, to select the optimal one for the specified metrics.

Institutional Repository - Library & Information Centre - University of Thessaly
30/12/2021 10:19:35 EET - 137.108.70.14

Abstract

Τα τελευταία χρόνια οι ανάγκες για περισσότερη υπολογιστική ισχύ, ταχύτερα μικροηλεκτρονικά κυκλώματα και με χαμηλή κατανάλωση ενέργειας έχουν οδηγήσει την βιομηχανία ημιαγωγών στην ανάγκη εύρεσης νέων μεθοδολογιών και τεχνολογιών, ώστε να καταστήσουν εφικτή την συνέχιση της κλιμάκωσης των ολοκληρωμένων κυκλωμάτων.

Ο νόμος του Moore, για τον διπλασιασμό των τρανζίστορ στην επιφάνεια ενός τρανζίστορ κάθε 18 μήνες [1], έχει φτάσει σε τέλμα. Ο λόγος είναι αφανώς ότι οι διαστάσεις των τρανζίστορ δεν μπορούν να μικρύνουν περισσότερο και αφετέρου ότι το κατασκευαστικό κόστος είναι πάρα πολύ μεγάλο. Παράλληλα, η καθυστέρηση των καλωδίων δεν μπορεί να κλιμακωθούν, όπως πραγματοποιείται με την καθυστέρηση των τρανζίστορ, εμποδίζοντας έτσι την κλιμάκωση των κυκλωμάτων.

Ένας φυσικός τρόπος για να επιτραπεί η αύξηση της πυκνότητας των τρανζίστορ, διατηρώντας σταθερή την επιφάνεια του ολοκληρωμένου κυκλώματος, είναι η χρήση πολλαπλών ορόφων. Όπως οι πολυκατοικίες επιτρέπουν περισσότερους ανθρώπους να διαμένουν στο ίδιο οικοδομικό τετράγωνο, σε σχέση με τις μονοκατοικίες, έτσι και τα τρισδιάστατα κυκλώματα μπορούν να επιτρέψουν την συνέχιση της κλιμάκωσης των ολοκληρωμένων κυκλωμάτων. Με την τρισδιάστατη σχεδίαση των κυκλωμάτων, δεν απαιτείται η χρήση νέων υλικών, αλλά ούτε και κατανάλωση οικονομικών πόρων για την περαιτέρω σμίκρυνση των τρανζίστορ. Έτσι, μπορούν να χρησιμοποιηθούν τεχνολογικοί κόμβοι που έχουν δοκιμαστεί και δεν εμφανίζουν προβλήματα λόγω των μικρών διαστάσεων.

Ωστόσο, παρόλο το γεγονός ότι η τρισδιάστατη σχεδίαση υπάρχει εδώ και δεκαετίες, δεν χρησιμοποιείται για μαζική παραγωγή κατά το έπακρον. Ο λόγος είναι ότι αφενός κατά την τρισδιάστατη σχεδίαση εμφανίστηκα προβλήματα, όπως η αύξηση της θερμοκρασίας του κυκλώματος, η κατασκευή και διαχείριση των συνδέσεων μεταξύ των διαφορετικών επιπέδων, αλλά αφετέρου η έλλειψη εργαλείων και ροών για την σχεδίαση των τρισδιάστατων ολοκληρωμένων κυκλωμάτων. Λόγω της έλλειψης εργαλείων, για την κατασκευή των τρισδιάστατων ολοκληρωμένων κυκλωμάτων χρησιμοποιούνται τα συμβατικά εργαλεία και οι ροές σχεδίασης δισδιάστατων κυκλωμάτων. Ωστόσο, τα συμβατικά εργαλεία σχεδίασης δισδιάστατων κυκλωμάτων, δεν μπορούν να λάβουν υπόψιν

Institutional Repository - Library & Information Centre - University of Thessaly
30/12/2021 10:19:35 EET - 137.108.70.14

τους πλήρως τα πλεονεκτήματα που επιφέρει η τρίτη διάσταση και έτσι πολλές φορές δεν μπορούν να χρησιμοποιηθούν αποτελεσματικά.

Η εν λόγω έρευνα έχει ως στόχο την εξερεύνηση και αξιολόγηση διαφορετικών μεθοδολογιών ανάθεσης των στοιχείων ενός κυκλώματος σε πολλαπλούς ορόφους. Η ανάθεση των στοιχείων αποτελεί ένα από τα πρώτα βήματα στην κατασκευή ενός τρισδιάστατου κυκλώματος, ενώ οι αποφάσεις που λαμβάνονται σε αυτό το στάδιο μπορεί να αποδειχτούν καταλυτικές για την ποιότητα του κυκλώματος. Απώτερος στόχος είναι η επίτευξη των καλύτερων δυνατών αποτελεσμάτων σε κατανάλωση ενέργειας, ταχύτητας και έκτασης του κυκλώματος, αλλά και το πλήθος των διασυνδέσεων μεταξύ των διαφορετικών επιπέδων.

Αρχικά, εξερευνήσαμε διαφορετικές μεθοδολογίες και αλγορίθμους ομαδοποίησης των στοιχείων του κυκλώματος για να απλοποιήσουμε την σχεδίαση με την τεχνική διαίρει και βασίλευε. Οι τεχνικές ομαδοποίησης είναι αρκετά σημαντικές καθώς επιτρέπουν την διαχείριση αρκετά μεγάλων κυκλωμάτων. Έτσι εξετάσαμε την τρισδιάστατη χωροθέτηση των ιεραρχικών στοιχείων του κυκλώματος με χρήση αλγορίθμου που έχει σαν αρχικό στόχο τον ισομοιρασμό των ιεραρχικών στοιχείων στα διαφορετικά επίπεδα. Παράλληλα αναπτύξαμε ένα νέο αλγόριθμο ομαδοποίησης στοιχείων, ώστε να δημιουργήσουμε πιο προσιτή, για την τρισδιάστατη σχεδίαση, τμηματοποίηση του αρχικού κυκλώματος.

Έπειτα, χρησιμοποιήθηκα και εξετάστηκαν αλγόριθμοι και μεθοδολογίες στο επίπεδο των πυλών. Στο επίπεδο των πυλών υπάρχει μεγαλύτερη ευελιξία για την βελτίωση των διαφόρων μετρικών (όπως κατανάλωση ενέργειας και χρονισμός). Παρόλα αυτά, η διαχείριση των σύγχρονων ολοκληρωμένων κυκλωμάτων, με δεκάδες εκατομμύρια στοιχεία, καθιστά πρόκληση την χρήση των μεθοδολογιών στο επίπεδο των πυλών. Έτσι, οι μεθοδολογίες που παρουσιάζουμε έχουν σκοπό την αποδοτική σχεδίαση μεγάλων τρισδιάστατων κυκλωμάτων.

Τελικά, οι διαφορετικές μεθοδολογίες, αλγόριθμοι και ροές συγκρίνονται στο επίπεδο της τοποθέτησης και διασύνδεσης, και τα αποτελέσματα αυτών αποδεικνύουν ότι η χρήση της κάθε μίας αποσκοπεί στη βελτιστοποίηση διαφορετικών μετρικών, ανάλογα με την κατασκευαστική τεχνολογία που έχει επιλέξει ο σχεδιαστής. Έτσι, παρουσιάσουμε τα θετικά και αρνητικά στοιχεία της κάθε μεθοδολογίας, ώστε να κατευθύνουμε τον σχεδιαστή στα επιθυμητά αποτελέσματα.

Institutional Repository - Library & Information Centre - University of Thessaly
30/12/2021 10:19:35 EET - 137.108.70.14

The dissertation of Nikolaos Sketopoulos is approved by:

Prof. Christos Sotiriou
Chairman, Dissertation Supervisor
Associate Professor in Electrical and Computer Engineering, University of Thessaly

Prof. Georgios Stamoulis
Member, Dissertation Committee
Professor in Electrical and Computer Engineering, University of Thessaly

Prof. Nestor Evmorfopoulos
Member, Dissertation Committee
Associate Professor in Electrical and Computer Engineering, University of Thessaly

Prof. Dimitrios Bargiotas
Member, Examination Committee
Associate Professor in Electrical and Computer Engineering, University of Thessaly

Prof. Vassilis Paliouras
Member, Examination Committee
Professor in Electrical and Computer Engineering, University of Patras

Prof. Mihalis Psarakis
Member, Examination Committee
Associate Professor in Informatics, University of Piraeus

Prof. Nikolaos Bellas
Member, Examination Committee
Professor in Electrical and Computer Engineering, University of Thessaly

Approval in: **01/09/2021**

Institutional Repository - Library & Information Centre - University of Thessaly
30/12/2021 10:19:35 EET - 137.108.70.14

Institutional Repository - Library & Information Centre - University of Thessaly
30/12/2021 10:19:35 EET - 137.108.70.14

Contents

List of Figures xv

List of Tables xvii

1 Introduction 1
- 1.1 Motivation and Structure Of The Dissertation . . . 5

2 Background 7
- 2.1 Physical Design Flow Overview . . . 7
  - 2.1.1 Floorplanning . . . 8
  - 2.1.2 Placement . . . 10
  - 2.1.3 Clustering . . . 14
- 2.2 3D Placement and Partitioning . . . 15

3 3D Flows Literature Review 17
- 3.1 Block Based 3D Integration . . . 17
  - 3.1.1 Floorplanning . . . 17
- 3.2 State of the Art Fine Grained 3D Integration Flows . . . 18
  - 3.2.1 Shrunk2D Flow . . . 18
  - 3.2.2 Derated2D Flow . . . 19
  - 3.2.3 Cascade2D Flow . . . 20
  - 3.2.4 Compact2D Flow . . . 21

4 Block-Based 3D IC Tier Assignment Methodologies 23
- 4.1 3D Floorplanning . . . 23
  - 4.1.1 Number Partitioning Problem . . . 24
  - 4.1.2 Proposed Area Balanced Modules Assignment . . . 25
  - 4.1.3 Floorplanning Per Tier . . . 27
- 4.2 METIS-Based Partitioning . . . 29

Institutional Repository - Library & Information Centre - University of Thessaly
30/12/2021 10:19:35 EET - 137.108.70.14

4.3 Clustering 30
4.3.1 Breadth Hyper-Edge Coarsening Algorithm 31
4.3.2 Hybrid Hyper-Edge Coarsening Algorithm 34
4.3.3 Clustering Summary 36

5 Fine-Grained 3D IC Tier Assignment **37**
5.1 Partitioning 37
5.1.1 Bin-Based FM Partitioning 38
5.2 3D Legalization 39
5.2.1 Unconstrained 3D Legalizer 40
5.2.2 Constrained 3D Legalizer 41

6 Experimental Flows and Results **45**
6.1 Experimental Flow 45
6.1.1 Fine-Grained Tier Assignment Flow 46
6.1.2 Block-Based Tier Assignment Flow 48
6.2 3D Legalization Methods 51
6.3 Metal Stacking 53
6.3.1 Metal Stack Comparison 54
6.4 3D IC Methodologies Comparison 56

7 Conclusions and Future Work **61**

Bibliography **65**

Institutional Repository - Library & Information Centre - University of Thessaly
30/12/2021 10:19:35 EET - 137.108.70.14

List of Figures

1.1 Interconnect and Gate Delay [2] 1
1.2 2D and 3D Interconnections 2
1.3 3D Stack Options [3] 3
1.4 Schematic representation of different 3D IC stacking options [4] 4
1.5 Face-to-Face and Face-to-Back wafer stacking process steps [5] 4

2.1 Physical Design Flow 8
2.2 2D Floorplanning Examples 9
2.3 3D Floorplanning Example [6] 9
2.4 Placement Techniques 11
2.5 Forces Representation 13
2.6 Overlapping Components 13
2.7 Density Bins and Spatial Forces 14
2.8 Overlap Free Placement 14
2.9 Multi-level Clustering-Partitioning 15

3.1 Shrunk2D Flow 19
3.2 Derated2D Flow 20
3.3 Cascade2D Flow 21
3.4 Compact2D Flow 22

4.1 Complete Karmarkar–Karp (CKK) algorithm tree with pruned nodes of a six-element set example [7] 24
4.2 Pruned Area Balance Partitioning Tree 26
4.3 3D Floorplanning Example 28
4.4 Floorplan Trees of Figure 4.3 Example 28
4.5 Fanout Clustering Order Example 30
4.6 Breadth Hyper-Edge Coarsening Algorithm Example 32
4.7 Hybrid Hyper-Edge Coarsening Algorithm Example 35

Institutional Repository - Library & Information Centre - University of Thessaly
30/12/2021 10:19:35 EET - 137.108.70.14

5.1 Flat Bin-Based FM (FBBFM) Approach Example 39
5.2 Legalize Cell to Sub-row 42

6.1 Examined Approaches Tree 45
6.2 Fine-Grained Tier Assignment Flow 46
6.3 3D Multiple IPOs Flow 48
6.4 Block-Based Experimental Flow 50
6.5 Metal Stacks and Tracks [8] 54
6.6 ldpc Benchmark Congestion Analysis 55
6.7 *ldpc* Benchmark Routing and 3D Tiers Layout using Single Metal Stack . . 55
6.8 Vias Per Net Comparison Results 56
6.9 Congestion Comparison Results 57
6.10 Power Comparison Results 58
6.11 Area Comparison Results 58
6.12 Wirelength Comparison Results 59
6.13 3D Vertical Vias Comparison Results 60
6.14 Tier Assignment Execution Time Comparison Results 60

Institutional Repository - Library & Information Centre - University of Thessaly
30/12/2021 10:19:35 EET - 137.108.70.14

List of Tables

1.1 Integration Process Steps 3

3.1 State of the Art Monolithic 3D Placement Approaches 18

4.1 Netlist Modules to be Partitioned 25
4.2 Balance Partitioning Lookup Table 27
4.3 Clustering Approaches and Nets Fanout Order Options Comparison 36

6.1 Single vs. Multiple IPO Impact to 3D Flows - Best Single/Multiple IPO Results 49
6.2 3D Legalization: Inter-tier Vias vs. 3D Wirelength Ratio 52
6.3 3D Legalization Schemes Comparison 53

Institutional Repository - Library & Information Centre - University of Thessaly
30/12/2021 10:19:35 EET - 137.108.70.14

Institutional Repository - Library & Information Centre - University of Thessaly
30/12/2021 10:19:35 EET - 137.108.70.14

Chapter 1

Introduction

It is a fact that Integrated Circuits (ICs) combine more and more functional blocks year after year, while their size decrease. As a result, conventional Two-Dimensional (2D) ICs' density and wire complexity grow substantially, resulting in longer interconnections. Moreover, despite the fact that smaller technology nodes present shorter gate delay, interconnect delay does not follow the same trend. As a result, in deep submicron technologies, interconnect delay dominates gate delay, while it remains a critical bottleneck of modern System-on-Chips (SoC).

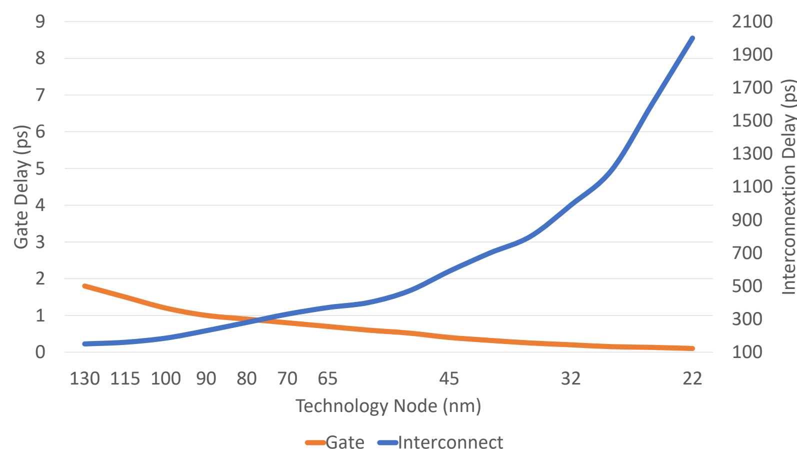


Figure 1.1 Interconnect and Gate Delay [2]

Figure 1.1 depicts interconnect and gate delay trends in correlation to various technology nodes. It is clear that due to the increasing interconnect delay, the total circuit delay is also increased at more resent technology nodes. Moreover, SoC incorporates multiple circuit types in one chip, such as logic, memory, analog/RF blocks, etc. Because different circuit families may require different substrates, this diverse integration in a single die becomes difficult.

Institutional Repository - Library & Information Centre - University of Thessaly
30/12/2021 10:19:35 EET - 137.108.70.14

Consequently, the semiconductor industry has began to seek new options due to the increase in power, performance, and cost barriers, specifically beyond 28 nm. Thus, three-dimensional integrated circuits (3D ICs) have been introduced to provide higher integration density and higher Power, Performance, and Area (PPA) gains.

Several studies have demonstrated that 3D ICs can result in 30% to 50% wirelength reductions [9–11] and correspond footprint area reductions (Figure 1.2). The reduction in wirelength leads to less power consumption in wires, buffer insertion reduction, and also increasing chip performance. The wirelength reduction occurs when long horizontal wires, *i.e.* wires on a 2D plane, become shorter by connecting cells vertically, as depicted in Figure 1.2.

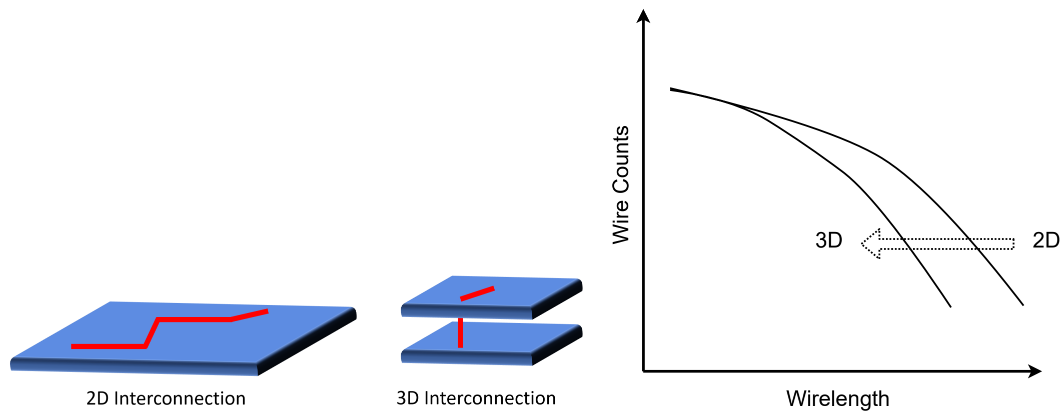


Figure 1.2 2D and 3D Interconnections

As a result, 3D integration is an up and coming technology that can assist the semiconductor industry in overcoming the challenges of shrinking technology nodes. To build integrated circuits in a vertical manner a number of options are available, as shown in Figure 1.3. However, this study focuses specifically on 3D ICs.

3D ICs integration technologies can be classified into two categories: sequential and parallel processes. In the case of a sequential process, the devices and metal layers of the second layer are developed on top of the first layer, etc. As a result, this integration approach is often termed as monolithic. The key technology in this method is the formation of a high-quality active layer that is separated from the bottom substrate. This bottom-up method has the advantage of establishing exact layer alignment. It does, however, have certain drawbacks. Top layer crystallinity is often low and imperfect [12], making high-performance device creation in the upper layers prohibitive. Furthermore, the high temperatures required to create the new top layers might harm the bottom layers. Monolithic integrated circuits are still in their early phases, and manufacturing performance is limited due to their sequential nature.

Institutional Repository - Library & Information Centre - University of Thessaly
30/12/2021 10:19:35 EET - 137.108.70.14

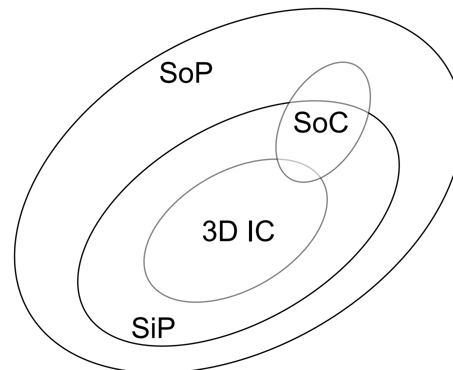


Figure 1.3 3D Stack Options [3]

The parallel process, on the other hand, employs standard production processes to handle each layer individually. These layers are then bonded together to form 3D IC by stacking each layer on top of the other ones. As parallel stacking doesn't involve a modification in the regular production process, it is more adaptable by the industry, compared to the monolithic approach. The parallel stacking can also be categorized as wafer-to-wafer, chip-to-wafer, or chip-to-chip based on the stacking method (Figure 1.4). In chip-to-chip stacking, wire bonding or Through Silicon Vias (TSVs) can be utilized to link chips vertically. TSV is the vertical connection utilized in wafer-level 3D integration, such as chip-to-wafer and wafer-to-wafer stacking, with the latter being the most cost-effective technique. On the other hand, chip-to-chip presents higher yield, since it allows the use of Known-good dice. Currently, R&D efforts are concentrated on chip-to-wafer integration.

Wafer thinning and handling, alignment, TSV creation, and bonding are all phases in this integration process, and the various options for each are summarized in Table 1.1.

Table 1.1 Integration Process Steps

Bonding Style	Metal-to-Metal	Dielectric-to-Dielectric	Hybrid Bonding
TSV Formation	Via First	Via Middle	Via Last
Stacking Orientation	Face-to-Face	Back-to-Face	Back-to-Back
Singulation Level	Chip-to-Chip	Chip-to-Wafer	Wafer-to-Wafer

An important choice for 3D integration is the stacking orientation of stack (Table 1.1). The two most important orientations are Face-to-Face and Face-to-Back. Figure 1.5 shows processing examples for these two stacking methods. At the latter orientation, the face of the

Institutional Repository - Library & Information Centre - University of Thessaly
30/12/2021 10:19:35 EET - 137.108.70.14

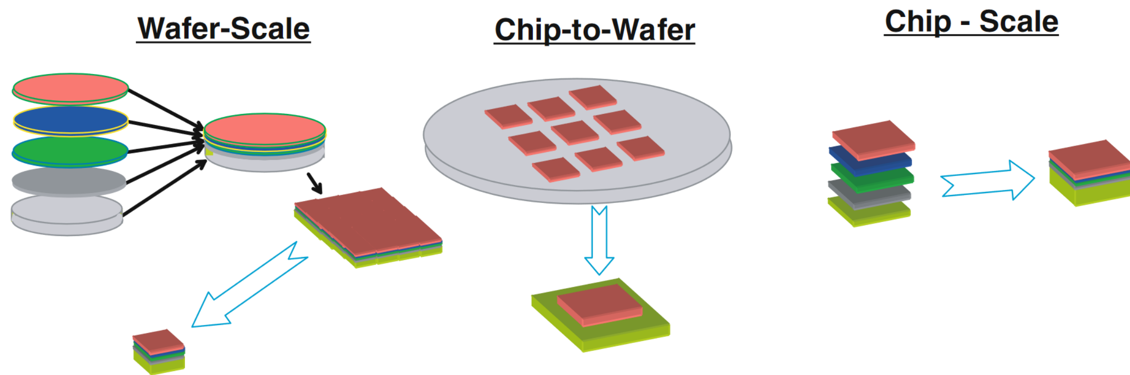


Figure 1.4 Schematic representation of different 3D IC stacking options [4]

bottom wafer is bonded to the backside of the top wafer. The top wafer is thinned from the backside using a handling wafer attached to the front-side after being prepped with TSVs. On the front-side of the bottom wafer, the necessary dielectric and metal are deposited before bonding with the backside of the top wafer. On the other hand, in the Face-to-Face approach, TSVs are fabricated on the front-side of the top wafer, while the top wafer is then bonded Face-to-Face with the bottom wafer.

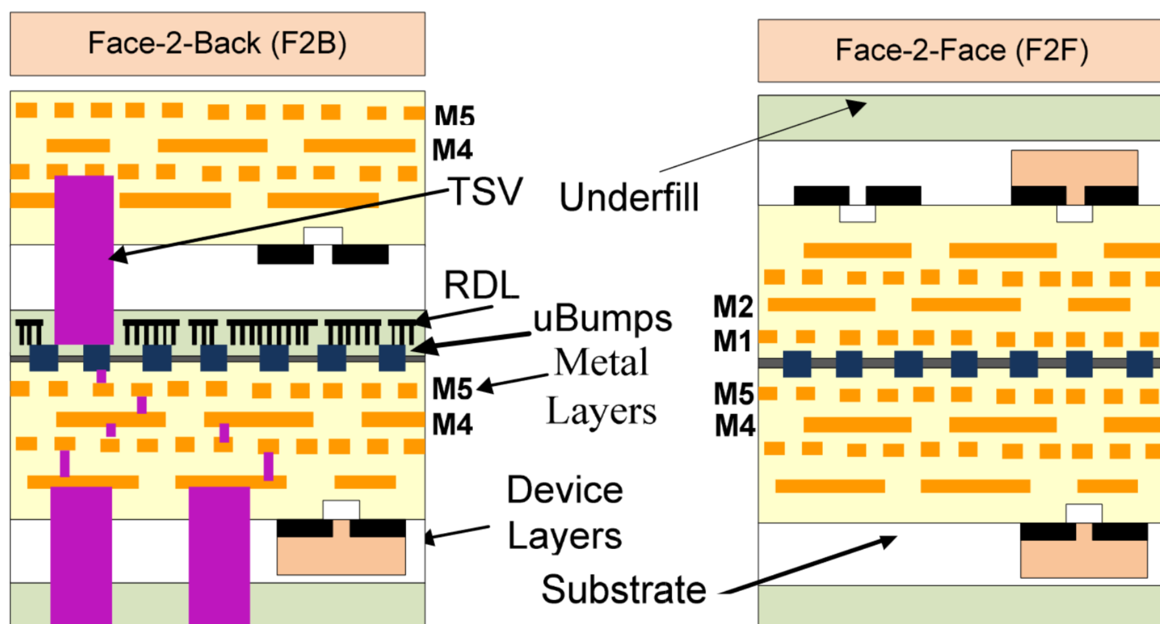


Figure 1.5 Face-to-Face and Face-to-Back wafer stacking process steps [5]

Apart from the integration process, the stack orientation, or the singulation level, it is important to have the appropriate EDA tools to design and test 3D ICs. The first attempts to manufacture 3D ICs used traditional 2D EDA tools, but yield was low. From the initial effort

Institutional Repository - Library & Information Centre - University of Thessaly
30/12/2021 10:19:35 EET - 137.108.70.14

to the present, EDA companies have made significant progress in providing a variety of 3D integration capabilities. Besides the fact that large EDA companies have released tools that enable 3D ICs features, 3D EDA tools are still in their early phases of development.

Synopsys, for example, introduced the "3DIC Compiler," a platform for advanced multi-die system design, implementation, validation, and signoff [13]. According to Cadence, "OrbitIO" is utilized for chiplet planning and managing system connectivity for complicated multi-fabric systems. Despite academic and commercial efforts to completely enable 3D integration, there are no complete solutions to all 3D IC levels and methods [14].

1.1 Motivation and Structure Of The Dissertation

Based on the growing interest of both academia and industry for the vertical chip integration as well as the shortage of 3D EDA tools, this research provides not only benefits but also disadvantages of distinct 3D integration approaches, techniques, and processes. The main topic of this research is physical design, and in particular to solve the problem of the tier assignment. Tier assignment can be defined as the problem to find a tier for all standard cells of an 3D IC, where final circuit PPA gains will be maximized. This selection affects numerous parameters, including the 3D IC's power, performance, and area (PPA), as well as several additional restrictions, such as the number of vertical interconnections. Due to the fact that assignment is a general-purpose problem, the proposed methodologies and algorithms are applicable to various integration process options, which are summarized in Table 1.1. However, some approaches may be more appropriate for a specific process option, as observed by the experimental results. Furthermore, by taking the limitations of each integration technique into account, the investigated methodologies may be employed for either monolithic or TSV-based 3D ICs. So, in this work we do not focus on a particular type of 3D IC.

Based on the granularity of the assignment problem, we investigate two primary strategies, the Block-Based and the Fine-Grained 3D IC assignment. For each of these strategies, we provide several solutions, either by leveraging external tools or by developing algorithms. Furthermore, we evaluate alternative design flows based on the techniques investigated, using an industrial tool for placement optimizations and routing.

The remaining part of this dissertation is organized as follows:

The physical design background for 2D and 3D ICs is presented in Chapter 2. Several procedures, strategies, and algorithms are presented briefly to introduce the key terminology that are utilized in the next chapters. First, we present the major floorplaning principles. Then, an overview of placement techniques is provided, followed by detailed descriptions

Institutional Repository - Library & Information Centre - University of Thessaly
30/12/2021 10:19:35 EET - 137.108.70.14

of the most important placement categories. Following that, we will cover clustering, an essential approach for reducing design complexity of very large circuits. Chapter 2 includes a brief overview of 3D placement strategies, while the most significant 3D floorplanning algorithms and placement flows are presented in Chapter 3.

The multiple Block-Based tier assignment techniques that we utilize in this research are presented in Chapter 4. First, we propose a 3D floorplanning algorithm with the goal of creating balanced levels. Then we employ the well-known multi-level partitioning tool, METIS, in two variants. The first one utilizes the tiers as partitions, thus the objective is to assign all the cells to tiers (partitions), by reducing vertical interconnections. The second is to construct multiple partitions, each of which is considered as a cell group (or cluster), while each group will be assigned to a tier.

Chapter 5 presents the Fine-Grained tier assignment methodologies. A bin-based partitioning methodology is presented, capable of handling very large designs. Moreover, the first 3D legalization algorithm is presented, with or without vertical connection constraints.

Chapter 6 presents the physical design flows that have been used to compile the experimental results for the investigated approaches. We also present the results of the investigated methods and algorithms, while we give recommendations for the better use of each methodology and algorithm. Finally, the conclusions and future work are presented in Chapter 6 as well.

Institutional Repository - Library & Information Centre - University of Thessaly
30/12/2021 10:19:35 EET - 137.108.70.14

Chapter 2

Background

This chapter provides a quick introduction of cutting-edge concepts for 2D and 3D physical design. The physical design flow is composed of multiple phases and sub-stages with strong connections between them (Figure 2.1).

2.1 Physical Design Flow Overview

The physical design flow consists of multiple stages, while some of the stages may be performed multiple times. In general, the flow starts from a synthesized netlist, extracted from a Register Transfer Level (RTL) code (either Verilog or VHDL). The first step of the physical design flow, is floorplanning, which determines the dimensions and the positions of the functional blocks in the chip area by minimizing a set of parameters, *e.g.* area (see Section 2.1.1). Furthermore, the top level I/Os pins are inserted at this step. The partitioning phase is optional but crucial for particularly big designs, while power planning defines the chip's power grid. The placement stage (see Section 2.1.2) is the most significant since it determines the position of standard cells, intellectual properties, blocks, and so on, by minimizing various sets of cost functions (*e.g.* wirelength, circuit delay, power, thermal and wire congestion). The placement stage is also essential because it has a global perspective of the circuit for all standard cells and macros, while it is responsible for effectively meeting the Power, Performance, and Area (PPA) requirements. The clock tree synthesis also specifies the clock network and provides buffers for more consistent delay sharing. Furthermore, routing is the stage at which the nets acquire physical form. Finally, additional optimizations and verification tests are performed to ensure that the chip's behavior is proper and that the various constraints are satisfied.

Institutional Repository - Library & Information Centre - University of Thessaly
30/12/2021 10:19:35 EET - 137.108.70.14

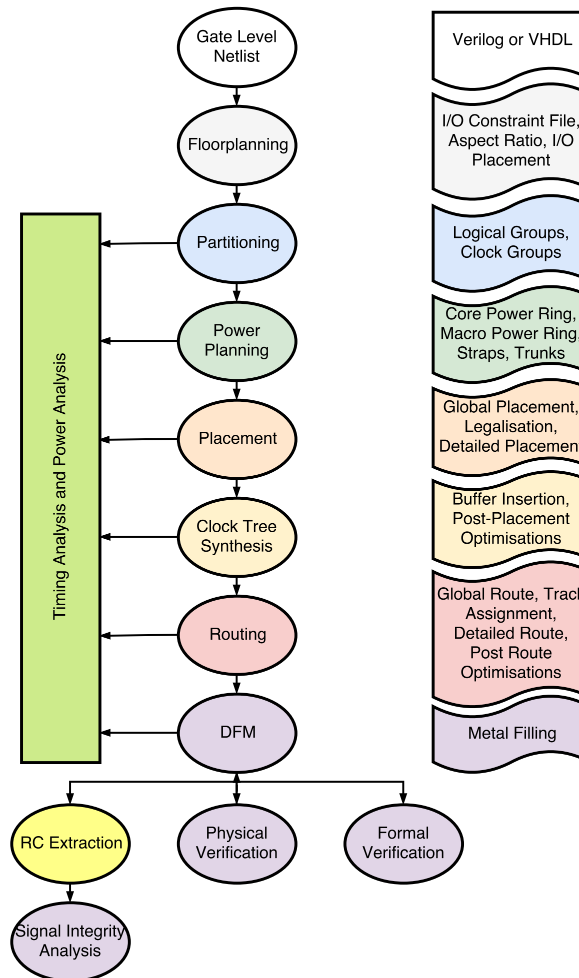


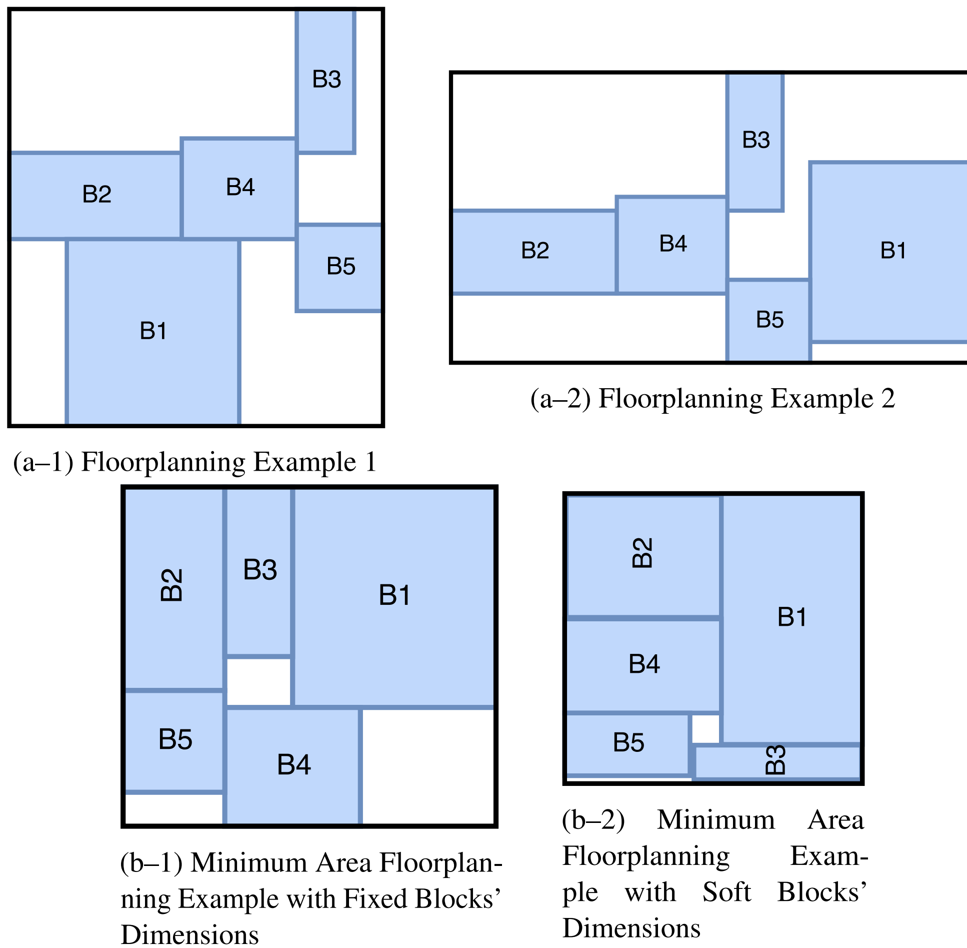
Figure 2.1 Physical Design Flow

2.1.1 Floorplanning

Floorplanning is an essential part of the physical design flow since it serves several purposes. By providing a collection of hard blocks (with fixed geometries) and/or soft blocks (whose geometries can be changed) and their connectivity, floorplanning determines the geometries of soft blocks and finds the locations of the blocks inside chip area, in order to optimize predefined cost metrics such as wirelength and chip area [15].

Floorplanning offers early feedback on the P&R stage since it is a pre-processing phase that recommends modifications, while it determines the core area and estimates circuit delay and wire congestion as well. Furthermore, floorplanning is utilized to hierarchically split a design into manageable smaller functional blocks, resulting in the reduction of the complexity in modern designs, due to its divide-and-conquer nature.

Institutional Repository - Library & Information Centre - University of Thessaly
30/12/2021 10:19:35 EET - 137.108.70.14



(a–1) Floorplanning Example 1

(a–2) Floorplanning Example 2

(b–1) Minimum Area Floorplanning Example with Fixed Blocks' Dimensions

(b–2) Minimum Area Floorplanning Example with Soft Blocks' Dimensions

Figure 2.2 2D Floorplanning Examples

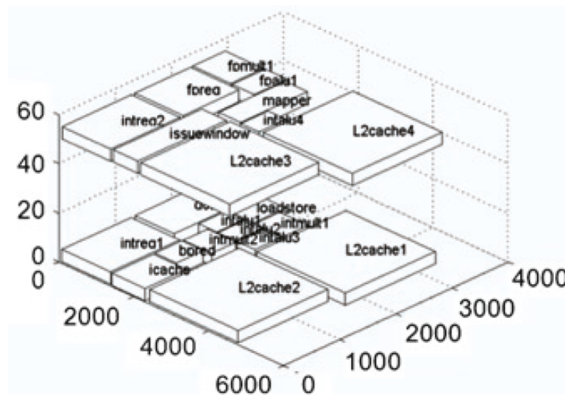


Figure 2.3 3D Floorplanning Example [6]

Floorplanning can be conducted for arbitrary dimensions, such as 1D, 2D, 3D, (Figures 2.2 and 2.3). Similar to conventional 2D floorplanning, 3D floorplanning likewise results in a compact packing area, short wirelength, low power consumption, and high performance [6].

Institutional Repository - Library & Information Centre - University of Thessaly
30/12/2021 10:19:35 EET - 137.108.70.14

3D floorplanning distributes blocks on several tiers without overlaps allowing design parameters such as chip area, wirelength, TSV number, and maximum on-chip temperature to be optimized or met.

The third dimension makes floorplanning a much more complex task since a multi-tier structure escalates solution space substantially, while higher density accentuates the thermal problem. As a result, shifting to 3D designs becomes a more challenging goal.

Most 2D and 3D floorplanning algorithms may be categorized as slicing or non-slicing, according to [16]. The most significant differences between 2D and 3D floorplanning are that the latter must manage an additional (third) dimension, while it must meet more constraints, such as thermal issues.

The third dimension can be handled by algorithms such as 3D transition closure graph, sequence triple, and 3D slicing tree [17] where the circuit blocks are notated by a set of 3D functional modules that determine the volume of the 3D system. By utilizing such a notation for the circuit blocks, an upper bound for 3D slicing floorplans is determined. In general, 3D floorplans result in larger unused space as compared to 2D slicing floorplans primarily due to the highly uneven volume of the 3D modules. For high flexibility ratios, however, this gap is reduced considerably, and, in certain cases, the upper bound for 3D floorplans becomes smaller than in 2D floorplans. For the second challenge of 3D floorplanning, the exploration of the solution space, a multi-level approach can often be more efficient for floorplanning 3D circuits than a flat approach, as discussed in the following subsections [16].

2.1.2 Placement

Following floorplanning, the next stage is placement, which is considered as one of the most significant and powerful steps in the physical design flow. Placement converts a block/gate/transistor-level netlist into an actual layout for timing convergence [18], *i.e.* it determines the exact locations of logic netlist components inside a particular area and establishes the design's overall timing characteristic. Furthermore, substantial optimizations are performed at this stage to optimize the timing of the design and to ensure a legal placement solution. As a result, the majority of industrial and academic physical design tools are developed around a placement infrastructure.

The typical objective of placement is to minimize its wirelength, and a fundamental constraint is to avoid any cell overlap. This is because wirelength can easily be modeled which serves as a good first-order approximation of real objective functions such as timing, power and routability of a design. There are three types of placement techniques, the stochastic, the partition-based, and the analytical, as presented in Figure 2.4. The first category uses randomization to optimize placement cost functions (*e.g.* TimberWolf [19]). Partition-based

Institutional Repository - Library & Information Centre - University of Thessaly
30/12/2021 10:19:35 EET - 137.108.70.14

placers, on the other hand, utilize a divide-and-conquer strategy to split the netlist and chip area into smaller regions [20–22]. Despite the fact that these approaches have short execution time, the solution may not converge to global optima. The third type is analytical placers, which are further split into two major sub-categories based on their objective function. These two categories are the non-linear and the quadratic. The former minimizes a non-linear cost function, *e.g.* a log-sum-exponential [23] or a weighted-average [24] function, which is minimized by non-linear optimization techniques like conjugate-gradient optimization [25]. While the objective function of the latter is quadratic and can therefore be minimized efficiently by solving a system of linear equations.

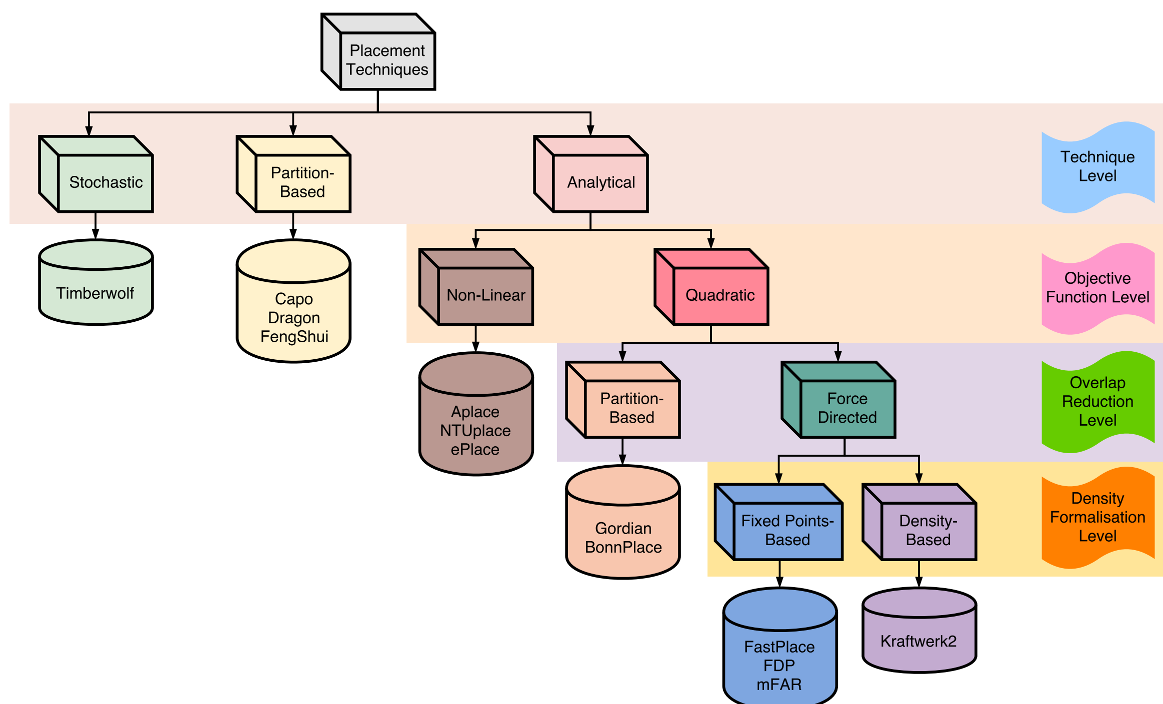


Figure 2.4 Placement Techniques

Analytical Placement Techniques

Analytical placement methods solve the placement problem as a single nonlinear or quadratic program, according to [18]. Instead of an accurate wirelength metric and exact non-overlapping constraints, the non-linear program uses wirelength approximations. The wire-length metric is a continuous differentiable function that is more complex than quadratic functions but more accurate in approximating HPWL. Moreover, density constraints are used to replace non-overlapping constraints in the placement region. The exact bin density function is a piece-wise linear function and hence not differentiable. The density of a bin

Institutional Repository - Library & Information Centre - University of Thessaly
30/12/2021 10:19:35 EET - 137.108.70.14

is a smooth version of the exact one. Examples of placers in this category are APlace [26], NTUplace [23], and ePlace [24]. To approximate the wirelength, APlace and NTUplace use the log-sum-exponential wirelength function, while ePlace uses the weighted average. To smooth the density function, APlace and NTUplace use a bell-shaped function proposed also by [27], and ePlace uses the local smoothness of the gradient function.

On the other hand, quadratic placers are often employed because they produce high-quality results while consuming little CPU time [28]. However, they encounter two challenges. First, they require a model to express a linear objective, such as a linear wirelength, in the quadratic objective function. Second, a method is required to decrease the module overlap that is common in quadratic placement. Quadratic placers are classified into two types based on the approach used to eliminate module overlap: (1) partition-based quadratic placers such as Gordian [29] and BonnPlace [30], which accomplish overlap-free placement using center-of-mass restrictions. These quadratic placers frequently partition the placement area recursively and assign modules to the placement partitions to improve the center-of-mass constraints. (2) Force-directed quadratic placers, such as Kraftwerk2 [31], FDP [32], and mFAR [33], use an additional force to distribute modules on the chip and therefore decrease module overlap. The additional force required for this category appears to be implemented in a variety of ways. For example, mFAR expresses the additional force using two distinct fixed points. One is to decrease module overlap, while the other is to establish force equilibrium. Kraftwerk2 calculates extra forces based on module density, which drives the modules from high to low-density regions.

Force Directed Placement

The force-directed quadratic placement algorithms imitate cell connections as springs, and their movement is governed by Hooke's law [34]. Cells are considered as points, regardless of their physical shapes or sizes, and each cell pin is located in a single point, usually near the cell's center. The single point model has the advantage of grouping wires that connect terminals of the same pair of cells. Because standard cells are small in comparison to the total placement area, this assumption is acceptable in standard cell design. Such an approximation cannot, however, be utilized in macro-cell placement as there is considerable error. In some cases the largest cell can occupy over half of the entire placement region, so approximating it as a point and gathering all its terminals to that point is obviously not efficient as it can affect the compactness of the final placement. As a result, the actual shapes and dimensions of the cells must be used to result in a more realistic solution. However, by utilizing the actual dimensions, overlaps between the cells will occur.

Institutional Repository - Library & Information Centre - University of Thessaly
30/12/2021 10:19:35 EET - 137.108.70.14

To solve the placement problem, force-directed placers model the components' connection as spring forces, while these forces are assigned to the netlist graph (Figure 2.5).

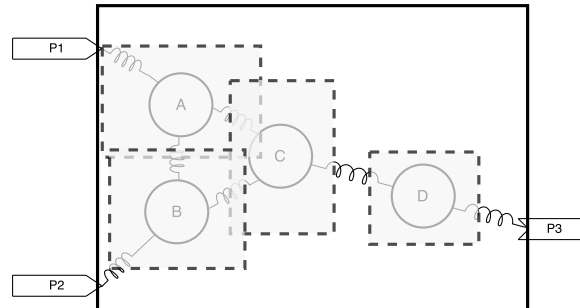


Figure 2.5 Forces Representation

Each graph node represents a component with physical dimensions (Figure 2.6). As a result, by minimizing only the wirelength, the placement solution will suffer from excessive high density, causing the router to produce poor results or even fail routing. Therefore, there are different approaches to spread components by optimizing not only wirelength, but also components density or overlaps. There are two ways to represent the overlap metric as an additional force: placing a pseudo-pin (*e.g.* [35]) or inserting additional forces based on the component density (*e.g.* [31]).

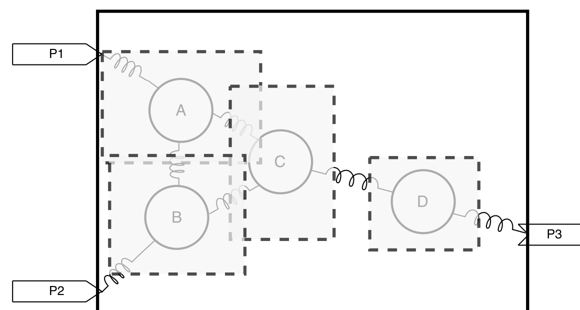


Figure 2.6 Overlapping Components

Components must be distributed evenly within the core area to avoid component overlaps. As a result, a two-dimensional bin building is formed in the layout area. Spatial forces are generated by transferring "supply" bins to "demand" bins. The density of each bin is defined as the ratio of the total area of the bin to the total area of the cell that overlaps with the bin. Additional forces are employed to transfer cells from supply to demand places to make the supply-demand system uniform, *i.e.* removing cell overlaps. Following cell spreading, the legalization process is conducted to completely eliminate cell overlap and place the cells on the placement grid.

Institutional Repository - Library & Information Centre - University of Thessaly
30/12/2021 10:19:35 EET - 137.108.70.14

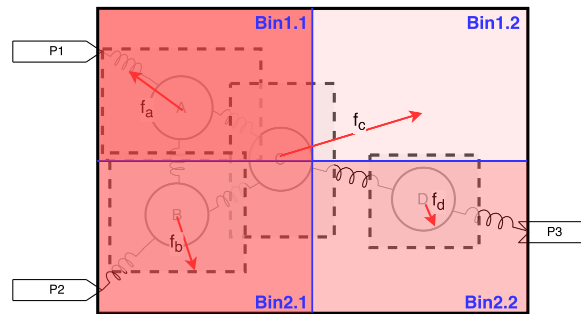


Figure 2.7 Density Bins and Spatial Forces

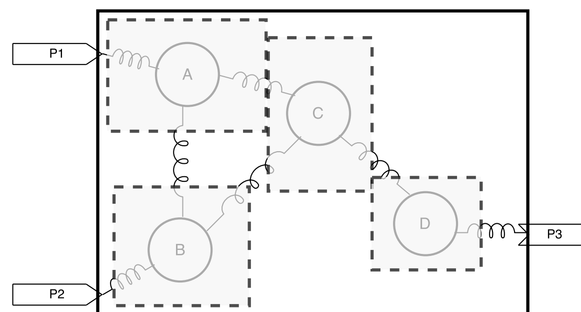


Figure 2.8 Overlap Free Placement

2.1.3 Clustering

The constant increasing amount of components in modern designs makes the placement process harder to solve. The large number of components has an impact not only on the quality of the results, but also on the execution time. Many placement techniques (for example, simulated annealing) become inefficient when dealing with large designs. To address this issue, a preparation phase that decreases the number of objects for placement while preserving placement quality is required. One widely used method is to form component groups (clusters) that decrease the amount of items to be handled by the placer. Clustering is a technique that contains various methods of extracting groups (*e.g.* First-Choice [36] and Best-Choice [37]). The selection of the components that form a cluster must be made very carefully, because it has impact on the quality of the placement. Clusters are made up of highly connected components, and depending on the clustering methods, these clusters may or may not be balanced. Clustering can enhance the quality of analytical placers particularly by 'hiding' the large fanout nets by clustering the components that belong to the same net. The size of each cluster, however, affects the behavior of the placer and hence affects placement quantity.

Modern placers use multi-level clustering and partitioning to reduce circuit complexity. Figure 2.9 depicts multi-level partitioning in which clustering is employed at various levels and subsequently clusters are assigned to partitions. The first stage is known as coarsening,

Institutional Repository - Library & Information Centre - University of Thessaly
30/12/2021 10:19:35 EET - 137.108.70.14

while the second is known as uncoarsening. The netlist or pre-placed components are grouped into tiny clusters during the coarsening process. Then, clusters from previous levels are grouped again to form new larger clusters of the current level. This technique is repeated until there are a few hundred clusters where the placer would be able to place them efficiently. Then, a partitioning algorithm assigns the clusters of the top level to partitions and then clusters are placed inside their partitions. After placing the top level clusters, the method proceeds by uncluttering the clusters from the previous level, and so on.

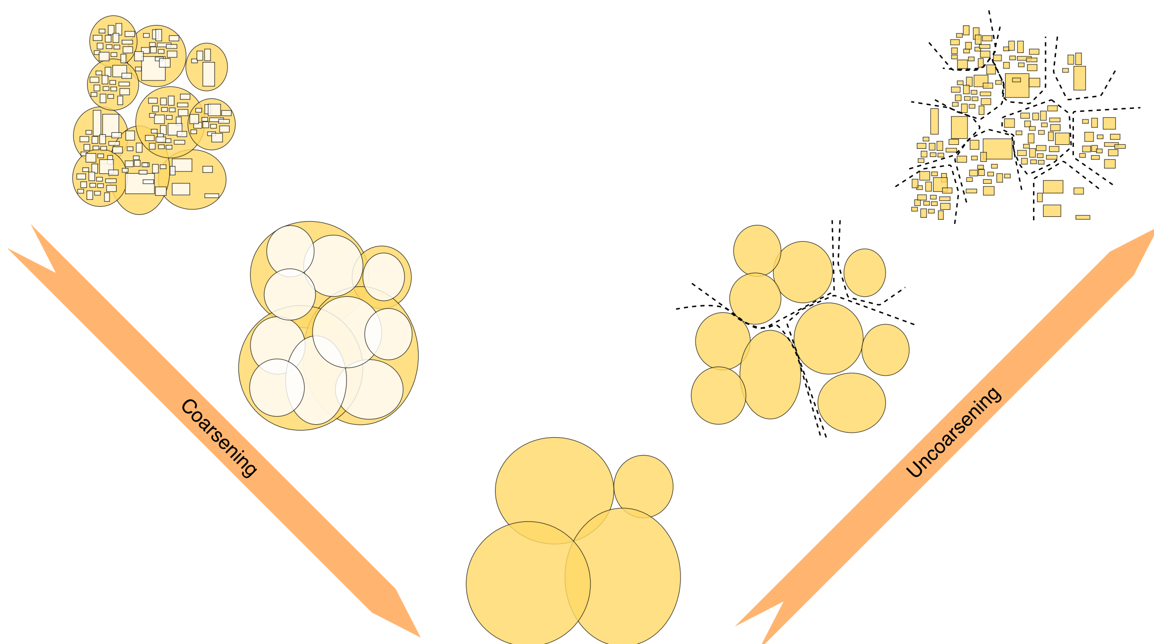


Figure 2.9 Multi-level Clustering-Partitioning

2.2 3D Placement and Partitioning

3D placement is an expansion of 2D placement that includes a vertical dimension. Placement for 3D ICs becomes considerably more difficult, similar to 3D floorplanning (Section 2.1.1). As with 2D placement, the primary goals of 3D placement are 1) wirelength minimization, 2) delay optimization, 3) thermal management, but it also considers 4) vertical interconnections [38]. Placement may also account for specific 3D stack properties, such as thermo-mechanical stress caused by Through Silicon Vias (TSVs). Currently, a variety of 3D placement approaches are also being developed to solve the problems associated with 3D IC technology, [6]. The majority of these methods, particularly at the global placement stage, can be thought of as expansions of 2D placement approaches. Partitioning-based techniques,

Institutional Repository - Library & Information Centre - University of Thessaly
30/12/2021 10:19:35 EET - 137.108.70.14

flat placement approaches, multi-level placement approaches, and folding-based approaches are four primary placement methodologies that can be used for 3D placement.

• The partitioning-based approach [39, 40] applies similar divide-and-conquer strategies as the well-known partitioning-based 2D placement approaches. The bi-section of the placement area in the z-direction is performed in parallel to the bi-sections in the x-direction and the y-direction. And the cost of partitioning is measured by the estimated wirelength and the TSV number, where the nets can be further weighed by thermal-aware or congestion-aware factors to consider temperature and routability.

• Flat placement approaches are the variations of quadratic placement, including the force-directed approach [41], the cell-shifting approach [42], and the quadratic uniformity modeling approach [43]. Since the unconstrained quadratic placement will introduce a great amount of cell overlaps, different variations are developed for overlap removal. The minimization of the quadratic wirelength, as well as the quadratic form of TSV number, could be transformed to the problem of solving a linear system. These flat placement approaches append a force vector (as previously described) to the right-hand side of the linear system, which is computed from the density distribution and helps to reduce cells overlaps. The vector is updated and the linear system is solved iteratively until the area in every predefined region is not greater than the area capacity of that region. These flat placement approaches differ by the definition of this force vector.

• The multi-level placement approach [44] constructs a physical hierarchy from the original netlist, and solves a sequence of Placement problems from the coarsest level to the finest level. An analytical 3D placement solver is applied at each level, which optimizes the log-sum-exp wirelength [45] and the log-sum-exp TSV number estimation subject to the overlap-free constraints. To model the 3D overlap-free constraints pseudo tiers are used for area projection, to guarantee the legality of the final solution.

• Folding-based placement reuses 2D placement solutions/tools and derives a 3D placement by folding modules and applying local refinements [38]. The main benefit is that proven 2D placers can be applied; the disadvantage is the limited consideration of implications arising from die stacking, *e.g.*, increased routing congestion.

Institutional Repository - Library & Information Centre - University of Thessaly
30/12/2021 10:19:35 EET - 137.108.70.14

Chapter 3

3D Flows Literature Review

3.1 Block Based 3D Integration

3.1.1 Floorplanning

Several works have been proposed for 3D integration within the floorplanning stage. Cong et al. [46] presented a thermally driven 3D floorplaning algorithm, which included thermal awareness selectively with a rapid but less precise hybrid resistive model and another precise but slow resistive model. They provide a trade-off between accuracy and runtime, which is reduced to 3.2x, when the less accurate model was used, compared to the accurate but slow model which results in approximately 10x runtime overhead.

In [47] authors proposed a thermal aware 3D floorplanning algorithm for microprocessors, which also takes interconnections into account. The algorithm identifies the thermal hotspots and utilizes power consumption in interconnections during the simulating annealing optimization step. By using the information about the interconnections of 2D floorplanning, they normalize the interconnections' wirelength of a 3D chip.

Healy et al. [48] have developed a multi-target micro-architectural floorplaning method that is used for 3D ICs. Taking thermal reliability, area, wirelength, vertical overlap, and bonding-aware partitioning into account, it defines the dimension and position of the functional modules by utilizing linear programming and utilizing simulated annealing technique.

For wirelength optimization, Ll et al. presented a hierarchical 3D floorplanning method [49]. The algorithm provides a statistical approach for partitioning and placing small groups of modules in each device layer, followed by 2D floorplanning in each device layer with no inter-layer movements. As a result, it divides the 3D floorplanning problem into many 2D floorplanning problems, where existing 2D floorplanning algorithms can be used.

Institutional Repository - Library & Information Centre - University of Thessaly
30/12/2021 10:19:35 EET - 137.108.70.14

Zhou et al. [50] employ a three-phase force optimization method along with legalizing approaches which remove block overlaps during floorplanning. This was the first work that used force-directed techniques for 3D IC floorplanning, compared to previous works that utilized the simulated annealing technique. A temperature-driven leakage model is also utilized to optimize the thermal profile and leakage power consumption based on a feedback loop [51].

3.2 State of the Art Fine Grained 3D Integration Flows

Since no commercial EDA tool that supports cell placement in 3D space is currently mature (there is an early endeavour with tools such as Innovus of Cadence [52] and IC Compiler of Synopsys [13]), several monolithic 3D and TSV-Based design methodologies which utilize 2D commercial EDA tools to implement Monolithic 3D designs are introduce [53–58]. In [53, 54], the authors propose design flows for gate-level monolithic 3D designs, whereas [57] presents block-level Monolithic 3D design flow. On the other hand, in [58] authors propose a TSV-Based methodology for gate-level designs. These methodologies achieve significant less wirelength results compared to 2D designs, reducing it from about 20% to 30%. Table 3.1 compares the flow of the four 3D implementation methodologies.

Table 3.1 State of the Art Monolithic 3D Placement Approaches

Property	Shrunk2D	Derated2D	Cascade2D	Compact2D
Granularity	Gate-Level	Gate-Level	Block-Level	Gate-Level
Tier Partitioning	Min-cut	Timing Slack	Architects Decide	Min-cut
Tier Interconnection	MIV Together	MIV Together	MIV Tier-by-Tier	MIV Together
Placement Optimization	Together	Together	Together	Together
Routing Optimization	Tier-by-Tier	Together	Together	Together

3.2.1 Shrunk2D Flow

As the z dimension is negligibly small, Shrunk2D flow (Figure 3.1) starts with scaling the x-y dimensions of all cells and metal layers by a factor of $(1/\sqrt{2})$. A Shrunk2D [53] design is implemented in half of its 2D equivalent footprint, by using shrunk cells. During the Shrunk2D design implementation, placement optimization is conducted, resulting in x-y location cells in the final monolithic 3D design. The Shrunk2D design cells are then scaled up to their original size, resulting in cell overlaps. Cell instances are then partitioned on the two tiers (*i.e.*, identify z position of cells) using an area-balanced min-cut partitioning method,

Institutional Repository - Library & Information Centre - University of Thessaly
30/12/2021 10:19:35 EET - 137.108.70.14

minimizing the number of vertical connections between the two tiers while balancing the cell area of the top and bottom tiers. The original metal stack is copied to establish the location of monolithic vias, creating the metal stack of the monolithic 3D design, i.e. both the metal stack on both the upper and the lower tier. Furthermore, the cells are labeled with their tiers, so that their pins are positioned on metal layers of the appropriate tier.

The design is routed using EDA tools with the complete metal stacks. The positioning of the monolithic vias depends on the position of the vias between the top metal layer of the lower level tier and the top tier's bottom metal layer.

Next, trial routing is performed on the netlists of each tier which uses the original design to achieve time constraints for each tier. Once the time constraints are determined, the timing driven routing is performed for each level, which leads to a final monolithic 3D design.

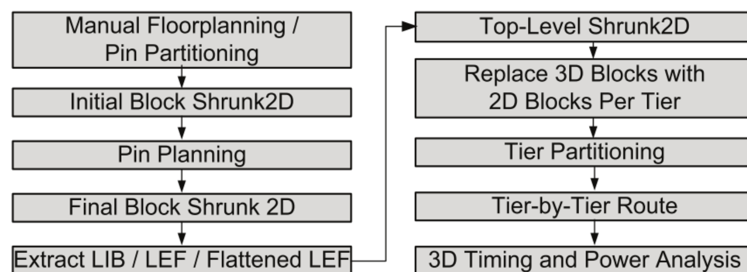


Figure 3.1 Shrunk2D Flow

3.2.2 Derated2D Flow

The monolithic 3D design flow given in [54] applies derived resistance and capacitance of wires, instead of utilizing shrunk cell models for three-dimensional placement since the shrinking cells are susceptible to DRC violation.

Derated2D starts from the 2D footprint, by utilizing the actual geometry of cells and wires. The footprint will be reduced by half at a later stage of the flow which will be described shortly. To force the P&R tool to embody with the third dimension, the resistance and capacitance of wires are derated by $1/\sqrt{2}$. The final Derated2D design's cell arrangement is then projected onto the Monolithic 3D design's footprint, by performing partitioning into two tiers. Due to the use of the derate factors, it is expected that the paracitics of the placement projection will be equevalent to that of the monolithic 3D design.

The remaining steps of the flow include vias planning, routing of each tier, timing optimizations and routing optimizations, shown in Figure 3.2.

Institutional Repository - Library & Information Centre - University of Thessaly
30/12/2021 10:19:35 EET - 137.108.70.14

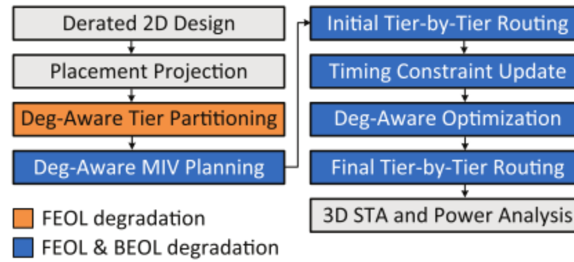


Figure 3.2 Derated2D Flow

3.2.3 Cascade2D Flow

In contrast to the previous two design flows, which implement two tier gate-level monolithic 3D ICs, Cascade2D flow [57] operates on functional blocks, that are divided into two tiers, implementing block-level monolithic 3D design. The flow (Figure 3.3) begins by separating RTL into two groups, one for the top and one for the bottom tier, which functional blocks must be set at the top and low tiers, respectively. The RTL partition is performed in two ways: (1) Based on the architectural structure of a design, a pair of modules with a limited time budget is selected and allocated to separate tiers, allowing them to profit from short vertical connections. (2) The number of time paths connecting to a couple of modules must be utilized in the absence of awareness of the design architecture. The first option completely delegated partitioning to the designer. If the first option is not practicable, the second option is selected.

Based on the findings of the 2D design, the authors first fix the critical paths on different tiers to reduce the distance between the cells that consist of these paths. The remaining cells are then partitioned to maximize the number of timing paths crossing the tiers while keeping tiers utilization balanced. Following tier assignment, the next stage is vias planning, which involves determining monolithic vias locations. This is done in a different fashion than in Shrunk2D and Compact2D. The top tier of Cascade2D is designed independently of the bottom tier, with monolithic vias port locations positioned on top of its appropriate receiving or driving cell to reduce the distance between monolithic vias and cell-pin. The bottom tier is then optimized separately using the vias port locations derived from the top tier. Then, a new die with both bottom and top tiers placed side-by-side is created with the same total area as the original 2D design.

Based on the locations of the monolithic vias, calculated in the previous step, the vias are placed at the top metal stack layer, while the layers below the top one is used to connect the monolithic vias to different tiers. In each partition, anchor cells are placed at the locations of vias, to be used as placeholders and bond the bottom metal layer of the top partition and top

Institutional Repository - Library & Information Centre - University of Thessaly
30/12/2021 10:19:35 EET - 137.108.70.14

metal layer of the bottom partition without delay emulating the behavior of monolithic vias. Finally, a transitional P&R flow is used for the entire design .

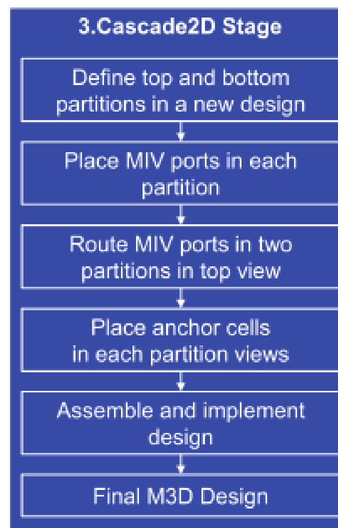


Figure 3.3 Cascade2D Flow

3.2.4 Compact2D Flow

Similar to Derated2D (Section 3.2.2), Compact2D [58] does not shrink the cells for the 3D placement. However, contrary to Derated2D, Compact2D is used for TSV-based placement and more specifically for Face-to-Face bonding as the vias are negligibly small. The most important advantage of TSV-based placement is the better correlation with the 2D tools and the process variation. For example, the authors of [58] present the industrial level users' view for Shrunk2D, reporting that the design rule violations have been increased exponentially, resulting many times in low-quality layouts. Compact2D, firstly places all the cells in the 2D floorplaned area and then shrinks the interconnect RC by $1/\sqrt{2}$, avoiding the redundant buffer insertions caused by the increased geometrical length. Then it introduces tier partitioning that utilizes bin-based placement-driven Fiduccia-Mettheyses [59] to assign each cell to a tier. High vertical interconnection leads to routing congestion, while low vertical interconnection decreases the PPA benefits of 3D ICs. Therefore, a sweet spot exists along the partitioning and the bin size. After partitioning, a legalization step is performed to legalize the cell of each tier and finally TSVs are placed based on the position of the cells.

Institutional Repository - Library & Information Centre - University of Thessaly
30/12/2021 10:19:35 EET - 137.108.70.14

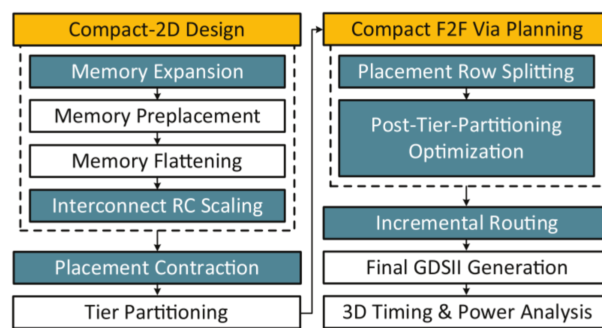


Figure 3.4 Compact2D Flow

Institutional Repository - Library & Information Centre - University of Thessaly
30/12/2021 10:19:35 EET - 137.108.70.14

Chapter 4

Block-Based 3D IC Tier Assignment Methodologies

Modern circuits, which contain billions of standard cells, lead circuit physical design to its limits. To enhance PPA gains, optimal positions and routing traces must be found for each standard cell. Moreover, similar to 2D integration, 3D integration is becoming increasingly difficult due to the complexity of contemporary circuits. As a result, circuit floorplanning and clustering approaches can be used to minimize circuit complexity by enabling higher levels of circuit abstraction. For the rest of the dissertation, the term Block-Based will be used to describe the methodologies that use blocks of cells, instead of flat standard cells.

4.1 3D Floorplanning

Many 3D Floorplanning algorithms strive to discover a solution to arrange functional blocks with the fewest number of inter-connections across tiers, the minimum wirelength, and the least thermal issues. However, area balancing across tiers is also essential since it might result in sub-optimal solutions by under-utilizing the die area. So, in this section, we will discuss the basic concepts of our 3D Floorplanning (3DF) algorithm, with the goal of reducing total (intra-tier and inter-tier) wirelength, the number of vertical connections, and tier area imbalance. Our 3DF algorithm can be divided into two main steps:

- Tier assignment of the functional modules to divide the circuit to equal size tiers
- Floorplanning of each tier to reduce the total wirelength.

Despite the fact that the 3DF algorithm can be applied at multiple tiers, it will be applied on two tiers, for fair comparison with other proposed approaches.

Institutional Repository - Library & Information Centre - University of Thessaly
30/12/2021 10:19:35 EET - 137.108.70.14

4.1.1 Number Partitioning Problem

Our 3DF algorithm's initial aim is to allocate netlist modules with the optimum tier area balance to fully use the core area. So, the first step is to determine the tiers' areas based on the areas of netlist modules. To estimate the tier area of the ideal area balancing tiers, we utilize the technique provided in [60]. The worst case complexity to solve the optimal balancing bi-partitioning problem is NP-complete, however there are various heuristics to minimize the complexity. Pruning heuristics are used to branch and bound the solution space without having any accuracy overhead.

In this section we describe a mathematical approach, proposed in [7], to find an optimal solution to the integer-partitioning problem. Then, we adopt this approach to evenly assign the module areas to tiers.

A binary tree structure is utilized to identify modules' assignment, while each branch represents a partitioning assignment. Figure 4.2, for example, depicts a binary tree containing six numbers, *i.e.* 6 5 5 4 3 3, ordered in decreasing fashion.

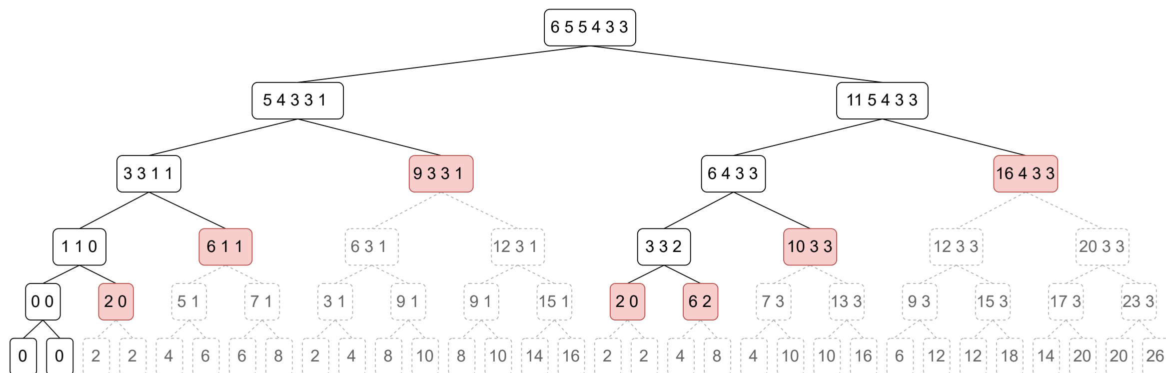


Figure 4.1 Complete Karmarkar–Karp (CKK) algorithm tree with pruned nodes of a six-element set example [7]

The left branch of each node indicates that the investigated numbers will be assigned to distinct partitions, while the right branch indicates that they will be assigned to the same partition. So, to build the root's left child node, we first replace the two greater numbers of the root node, and then sort the result, *i.e.* 1, and the remaining numbers of the root node that have not yet been assigned to a partition, *i.e.* 5, 4, 3 and 3. In the case of the right child, the root's two grater numbers are assigned in the same partition, thus their values are summed, *i.e.* 11. This method is used until there are no numbers to assign to partitions, i.e. only leaf nodes remained.

The complexity of this technique is $O(2^n)$ once all the nodes for the binary tree are explored. The branch-and-bound technique, on the other hand, can be used to trim branches

Institutional Repository - Library & Information Centre - University of Thessaly
30/12/2021 10:19:35 EET - 137.108.70.14

during exploration. For example, if a node's first value is greater than or equal to the sum of the remaining numbers, the tree node will be pruned since no better solution can be found by examining the remaining branch nodes.

So, in Figure 4.2, the red nodes indicate where the pruning heuristics were applied, and the dashed nodes indicates where the nodes were trimmed. This figure demonstrates how pruning algorithms can minimize the size of the binary tree.

4.1.2 Proposed Area Balanced Modules Assignment

Floorplanning in physical design defines the relative locations of modules on the chip depending on many criteria such as connectivity, area occupation, and aspect ratio [18]. Floorplanning algorithms can serve as a guideline in the placement process, reducing the design's complexity. Thus, floorplanning is critical since it impacts the whole physical design procedure. In this section we present how we applied the partitioning approach described in Section 4.1 to our 3D Floorplanning (3DFP) algorithm for two-tier 3D ICs with best area balance.

Area Balance Tier Assignmenet

Our 3DFP algorithm's major objective is to allocate netlist modules to tiers with equal area. As a result, the first step is to compute the tier areas based on the netlist module areas, while the second is to find the best modules' arrangement. We demonstrate our 3DFP with a simple example, where the names of the submodules and their regions are provided in Table 4.1.

Table 4.1 Netlist Modules to be Partitioned

Submodule Name	Total Components Area (um^2)
A	446.39
B	273.20
C	216.85
D	128.56
E	49.46
F	21.07
G	14.83

The first step is to construct a binary tree (Figure 4.2) in which each node has the necessary information to choose the optimal modules for tier assignment. Starting with the root node, where all the modules are listed as members of the node in decreasing area order, we choose the two larger modules, *i.e.* A and B (similar to the Number Partitioning Problem).

Institutional Repository - Library & Information Centre - University of Thessaly
30/12/2021 10:19:35 EET - 137.108.70.14

Nodes 2 and 3 are then formed. A new H member is added and sorted at the left node, *i.e.* node 2, based on the area difference between root members A and B.

As in [60], the left child nodes indicate that the examined members of the parent node are assigned to different tiers, so a mock-up member with area equal to the difference of the examined members is created. On the other hand, for the right child node, *e.g.* node 3, a new mock-up member is inserted. The area of the new member is equal to the sum area of the examined members, indicating that the two members have been assigned to the same tier. This procedure will be continued until the examined node does not have any members.

As described in Section 4.1, several heuristics [60] can be used to reduce the size of the the tree. For example, at node 3 the area of the member I (719.59) is greater than the sum area of the other members of the node, *i.e.* C, D, E, F, G (431.37). So, it is guaranteed that even if we create the entire tree, there is no combination of its members that leads to better balance assignment. So, there is no need to create left and right branches from this node. As a result, in Figure 4.2 the terminal nodes are: Node 3, 5, 9, 17, 32, 64 and 65.

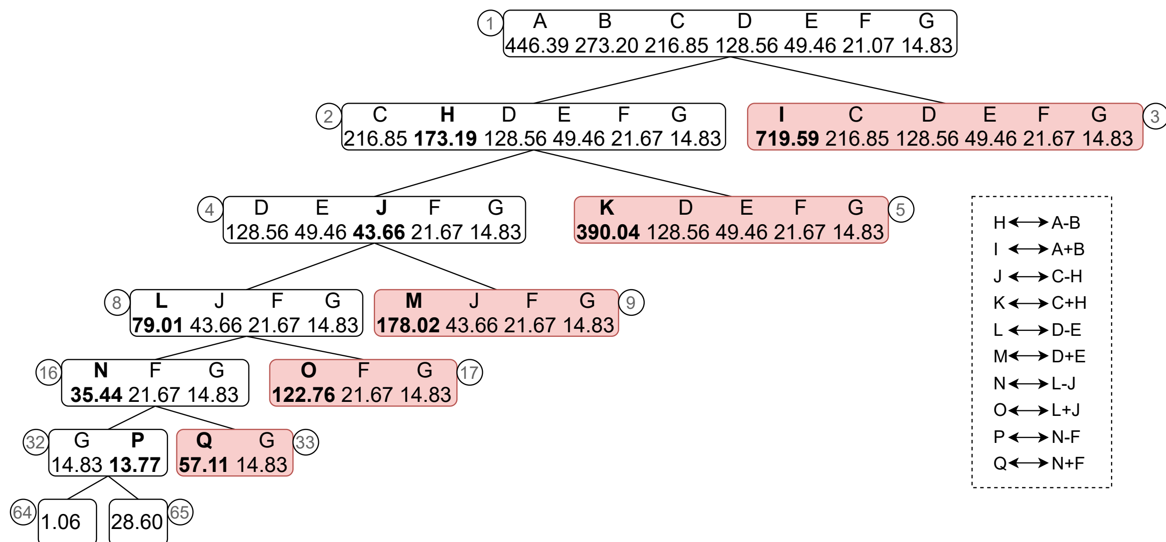


Figure 4.2 Pruned Area Balance Partitioning Tree

When the tree creation is finished, we have to find the tier assignment with the best area balance, between the terminal nodes. As in the example presented in Figure 4.2, the node with the best area balance is Node 64, where the difference between the tiers' area is $1.06um^2$. Now, starting from the node with the best area balance, a backward traversal is performed to assign all modules of the root node to tiers. Table 4.2 presents the members, of the node where they have been created (in parenthesis) and their tier assignment. Since Node 64 is a left child and has the smallest area difference, which means that the first two members, G and P, of the parent node, *i.e.* Node 32, are assigned to different tiers. So, lets

Institutional Repository - Library & Information Centre - University of Thessaly
30/12/2021 10:19:35 EET - 137.108.70.14

assume that member G is assigned to the top tier and member P is assigned to the bottom tier. However, member G is an actual netlist module, while member P is not. So, mock-up member P is replaced with the members that part it, *i.e.* the N and F. Figure 4.2 present a dashed legend, that correlates the mock-up members with the members that created them. As a result, member P has been created with the assignment of members N and F to different tiers. In this way, member P is replaced by member N and F, while F is assigned to the other tier, *i.e.* top tier. Similarly, member N is replaced with member L and J is assigned to the top tier. To find the member to replace, as fast as possible, we use the node where the member had been created. For example, at tree depth 3, Table 4.2, the node with the greatest index is replaced first, *i.e.* member L with index 8. The final tier assignment, at tree depth 0, will consist of only actual netlist modules. So, modules G, F, C, E and B are assigned to the top tier, while modules D and A are assigned to the bottom tier.

Table 4.2 Balance Partitioning Lookup Table

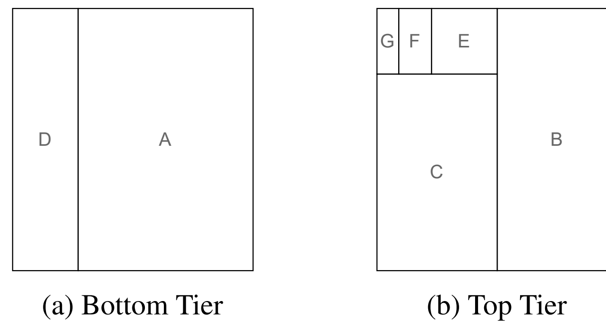
Tree Depth	Top Tier	Bottom Tier
5	G (1)	P (32)
4	G (1), F (1)	N (16)
3	G (1), F (1), J (4)	L (8)
2	G (1), F (1), J (4), E (1)	D (1)
1	G (1), F (1), C (1), E (1)	D (1), H (2)
0	G (1), F (1), C (1), E (1), B(1)	D (1), A (1)

4.1.3 Floorplanning Per Tier

The next step, after the modules tier assignment, is to perform floorplanning. In this work, we use a traditional 2D min-cut floorplanning algorithm per tier, similar to [61], to reduce the total wirelength. Due to the fact that the first step of our 3D FP approach evenly divide the modules to tiers, based on their areas, it is important to keep the modules to the tier that have been assigned. So, modules are not to change tier. Figure 4.3 presents the final floorplanning of the bottom and top tiers. Since the area of the tiers in most cases cannot be equal, *i.e.* there is no perfect area balancing, the final core area of each tier will be defined by the largest tier.

Then, the modules that have been assigned to the current tier are placed, based on an 2D min-cut floorplanning algorithm, similar to [61]. After the initial floorplanning, the optimization step will further reduce the total wirelength, through a Simulating Annealing (SA) process [62]. SA process allows escaping from local minimums to find better solutions till the temperature is high.

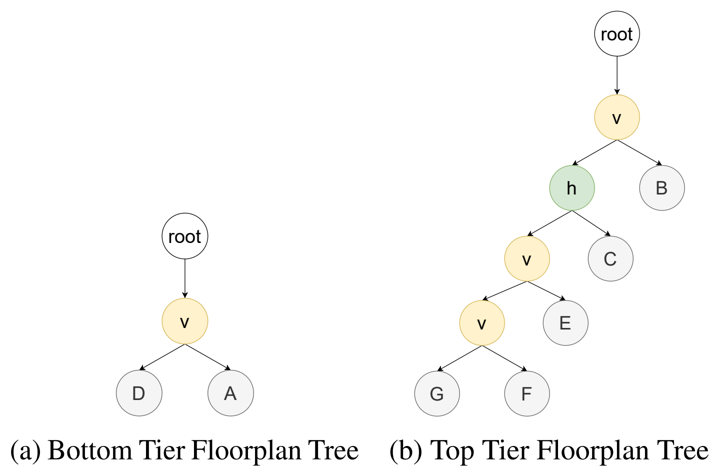
Institutional Repository - Library & Information Centre - University of Thessaly
30/12/2021 10:19:35 EET - 137.108.70.14



(a) Bottom Tier (b) Top Tier

Figure 4.3 3D Floorplanning Example

Our algorithm is capable of swapping and moving nodes (*i.e.* modules or groups of modules) of the floorplan tree, while it is able to change the cut type (vertical or horizontal) of node. Figure 4.4 illustrates the floorplan trees, of the bottom and top tier floorplans, depicted in Figure 4.3.



(a) Bottom Tier Floorplan Tree (b) Top Tier Floorplan Tree

Figure 4.4 Floorplan Trees of Figure 4.3 Example

Institutional Repository - Library & Information Centre - University of Thessaly
30/12/2021 10:19:35 EET - 137.108.70.14

4.2 METIS-Based Partitioning

As previously mentioned, it is very difficult to handle very large modern circuits in a flat way. Floorplanning is a very good solution that can be used as a divide-and-conquer technique to reduce the circuit complexity. However, floorplanning is more related to the functionality of the SoC. For example, CPU, modem, and DSP are some functional blocks in a modern SoC, that are developed independently from different groups in most cases, while they are connected together to form a SoC which are used as blocks in the floorplanning process. Besides floorplanning, we investigate an alternative solution to handle large designs, without relying on designer interference to separate SoC in functional modules. The idea is to extract the circuit modularity automatically and then perform floorplaning or placement on the extracted blocks.

In this concept, we first use the well-established multi-level partitioning tool METIS [63, 64] that is able to partition a large circuit very fast, by using coarsening and uncoarsening phases. In the coarsening phase, clustering is performed into several levels, while in the uncoarsening phase, the clusters are assigned to partitions till all the standard cells are assigned to a partition. METIS provides several features that can be used for the 3D integration. First of all, it aims to minimize cutsize, *i.e.* the number of connections between the partitions, while it handles area balance constraints.

In this work, we utilize two different flavors of METIS to

- partition the circuit into two tiers (hMETIS) [64],
- create clusters that are as floorplan modules (kMETIS) [36].

The former is used as a Fine-Grained tier assignment methodology in the experimental section, to assign the flat cells in tiers. On the other hand, the latter flavor is used to create new modules, instead of using netlist functional modules as mentioned in Section 4.1. As a result, the design hierarchy does not influence the modules assignment to tiers. This is important, because it is feasible to assign each cluster of cells without considering the hierarchy created by the designers, which may be sub-optimal for 3D ICs.

However, it is also important to notice that METIS is a closed source tool, that does not provide information for the clusters that are created, but only for the partitions. As a result, to divide a large design into blocks, we use the final partitions as clusters, by using the kMETIS flavor. More specifically, kMETIS is a multi-level hypergraph partitioning algorithm, that is able to create multiple partitions simultaneously, *i.e.* without performing bi-partitioning iteratively.

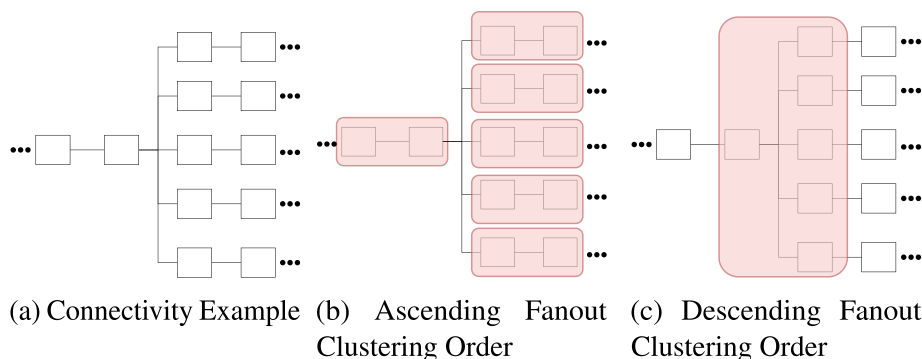
Institutional Repository - Library & Information Centre - University of Thessaly
30/12/2021 10:19:35 EET - 137.108.70.14

4.3 Clustering

As described in the previous section, METIS can be used to assign each standard cell to a tier to form a 3D IC with few vertical connections, while it provides area balanced tiers. However, due to the fact that the tool is a closed source, it is not easy to adjust different costs, or to force the tool to meet custom constrains. Thus, we developed a clustering algorithm similar to the Hyper-Edge Coarsening (HEC) phase of METIS [64], to explore the clustering benefits on 3D integration.

Starting from a synthesized netlist, our clustering algorithm first sorts the nets based on their fanout. Compared to the original version of HEC [64], that sorts the nets in ascending order, we investigate both ascending and descending orders. Ascending order can be used to reduce vertical connections, while descending order can be used to reduce the impact of high fanout nets on timing by providing higher priority to cluster the cells, of a high fanout net, together.

More specifically, modern circuit complexity leads to increased connectivity between the cells. As a result, multiple cells are connected to the same nets (specifically when buffering is insufficient). High connectivity influences both placement and routing (P&R), since connectivity has direct impact on the final placement and is related to the complexity of the P&R process. Moreover, the complexity of the partitioning algorithms like Fiduccia–Mattheyses [65], commonly used for 3D integration, is related to the number of nets and the number of cells/objects. So, by reducing the number of nets or the members of the cells that belong to the same nets (*i.e.* fanout degree), the partitioning algorithms can result in faster solutions, without significant impact on the quality of the results. As a consequence, multi-level partitioning algorithms, just like [64], can be applied to very large modern designs.



(a) Connectivity Example (b) Ascending Fanout Clustering Order (c) Descending Fanout Clustering Order

Figure 4.5 Fanout Clustering Order Example

Figure 4.5 presents a simple example of ascending and descending fanout order clustering. In ascending order, standard cells, with low fanout are grouped first, by reducing total number

Institutional Repository - Library & Information Centre - University of Thessaly
30/12/2021 10:19:35 EET - 137.108.70.14

of objects from 12 to 6 (Figure 4.5b), at the fist clustering level. However, the connectivity of the larger fanout net has not been reduced, so the clusters will be strongly connected. On the contrary, the descending order provides not only less objects, but also less connections. For example, in Figure 4.5c, the cluster absorbs all the connections that belong to the large fanout net, while the final number of objects will be 7.

Additionally, the connectivity of high fanout nets also slow down the placement step. Most of the state-of-the-art placers solve the placement problem as a convex minimization problem $Ax = b$, where A is the adjacency matrix of the circuit, b is a vector with the positions of the fixed cells and top-level ports, and x is the positions of the cells that lead to the minimum wirelength. At this point, it is important to point out that for large designs the size of the adjacency matrix is very large. But, the most important issue is that high fanout nets influence sparsity of the matrix. So, in the case where cells that belong to the same high fanout net are clustered together, the adjacency matrix becomes sparse enough to efficiently solve the $Ax = b$ problem, without any significant accuracy loss. Apart from cells' connectivity, the clustering algorithm must take either soft or hard area ratio constraints between the clusters into account. In the case of extremely unbalanced clusters, the physical design algorithms, such as placement, do not function properly.

Our multi-level clustering implementation aims to create groups of standard cells to reduce interconnection density and the total number of objects to speed up either the partitioning or placement process, while the clusters' area ratio meet the users constraints. Clusters' formation is achieved in two ways, either by clustering first the cells from the same net or by different nets that belong to a netlist path. The next sections describes these two clustering algorithms, that build the clusters in either breadth or hybrid fashion.

4.3.1 Breadth Hyper-Edge Coarsening Algorithm

In this section we present the Breadth Hyper-Edge Coarsening (BHEC) algorithm, that aims to create clusters from the same net per level, to reduce the connectivity or density of the circuit.

This algorithm performs clustering in multiple levels by taking the area ratio of the clusters into consideration. Figure 4.6 presents a simple example of the algorithm. For simplicity, there are three parts of a larger design containing 30 standard cells, as presented in Figure 5.1a, while the three parts are unconnected at the current level of abstraction. Additionally, the cells' areas vary, so for example *Cell 11* is 8 times larger than the area of *Cell 1*, while *Cell 21* is *2* times larger than *Cell 1*. To guarantee the area ratio of clusters, this algorithm takes the cells' areas into account.

Institutional Repository - Library & Information Centre - University of Thessaly
30/12/2021 10:19:35 EET - 137.108.70.14

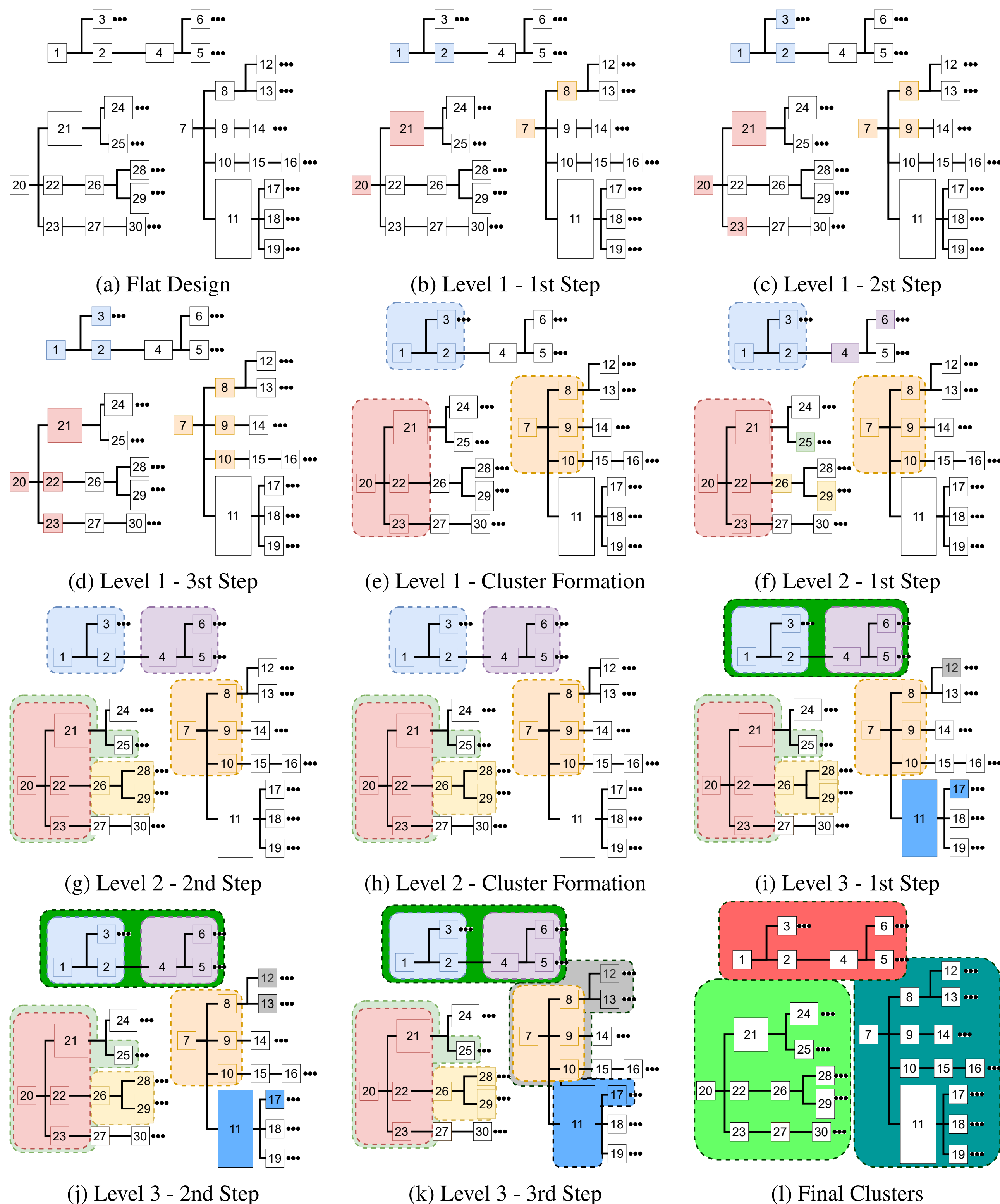


Figure 4.6 Breadth Hyper-Edge Coarsening Algorithm Example

Institutional Repository - Library & Information Centre - University of Thessaly
30/12/2021 10:19:35 EET - 137.108.70.14

In this example, we aim to build three clusters with maximum 2.5 area ratio. The algorithm sorts the nets based on their fanout, and the loading cells based on their "importance". The importance of the loading cells can be their size, their delay, etc., but for the purpose of this example the importance of all the cells will be the same, and so their selection is arbitrary.

Moreover, to generate clusters with guaranteed area ratio, the clusters grow by one cell at a time except for the 1st step of each level, where two cells are selected to create a cluster. This is because we assume that each cluster will have a minimum of two members. Figure 5.1b presents the first step of the first level, where the three clusters have been created (red, orange and blue) and each one contains 2 cells. However, despite the fact that the nets are sorted in decreasing order, in term of their fanout, the net that it is driven by *Cell 11* is not part of a cluster. This occurs due to the fact that *Cell 11* is large enough and will lead to area ratio violation. More specifically, if the blue cluster contains *Cells 1* and *2*, and the red cluster *Cells 20* and *21*, then their size will be 2 and 3. Based on the target area ratio, *i.e.* 2.5, the largest cluster can be 5, *i.e.* 2.5 times larger than the smaller cluster of the level. As a result, *Cell 11* cannot be part of a cluster in this step, and so its driving net is ignored in this step. Finally, the third cluster (the orange one) is created from the next net in sorted list, while it will contain *Cell 7* and 8, with total size 2.

When the required number of clusters is created, the clusters grow by inserting cells that belong to the common nets with the clustered cells. So, at the 2nd step, a new unclustered cell is grouped at each cluster. At this stage, all of the cells that belong to the net driven by *Cell 1* are clustered, and so this cluster will not grow any further at this level. So, our BHEC algorithm stops growing the blue cluster. Then, the other two clusters will continue to grow till there are either no unclustered cells in the starting net, or the clusters' area will reach the upper bound area ratio. In step 3, *Cell 22* and *Cell 10* are assigned to the red and orange clusters respectively. Since, the net driven by *Cell 20* is entirely part of the red cluster, this cluster cannot grow any more, while the only cell that can be assigned to the orange cluster is *Cell 11*. However, this cell is grouped together with the orange group, the size of the latter will be 10, and as a result it violates the clusters' area ratio. In this case, the first level of the BHEC algorithm stops, and the second level starts based on the already created clusters and the remaining unclustered cells.

Based on the nets fanout order, the first unclustered cell is *Cell 11*, but starting from this cluster the target area ratio will be violated. So, lets assume that the next candidate net is the net driven by *Cell 26*. Starting from that net, the yellow cluster with area equal to 2.5 will define the maximum cluster area as 6.25, at this step. Moving on to the next steps, the algorithm creates the light green cluster (area: 6), pink cluster (area: 3.5), and yellow cluster

Institutional Repository - Library & Information Centre - University of Thessaly
30/12/2021 10:19:35 EET - 137.108.70.14

(area: 3.5). Since the size of the clusters has been increased, *Cell 11* will be clustered in the next level to form the blue cluster, presented in Figure 4.6i.

Finally, Figure 4.6l presents the final level of clusters that guarantees the area ratio. It is important to note that this is a mock-up example to intuitively describe the algorithm, while several corner cases that have been handled are not presented.

4.3.2 Hybrid Hyper-Edge Coarsening Algorithm

A key disadvantage of the Breadth Hyper-Edge Coarsening algorithm is that the levels depend on the net fanout, *i.e.* clustering at this level stops when all the cells that part the net have been clustered. As the average fanout in a flat circuit is approximately 3-4, this algorithm leads to small, in terms of members, clusters. Moreover, small clusters lead to more levels of clustering, while more levels mean higher runtime.

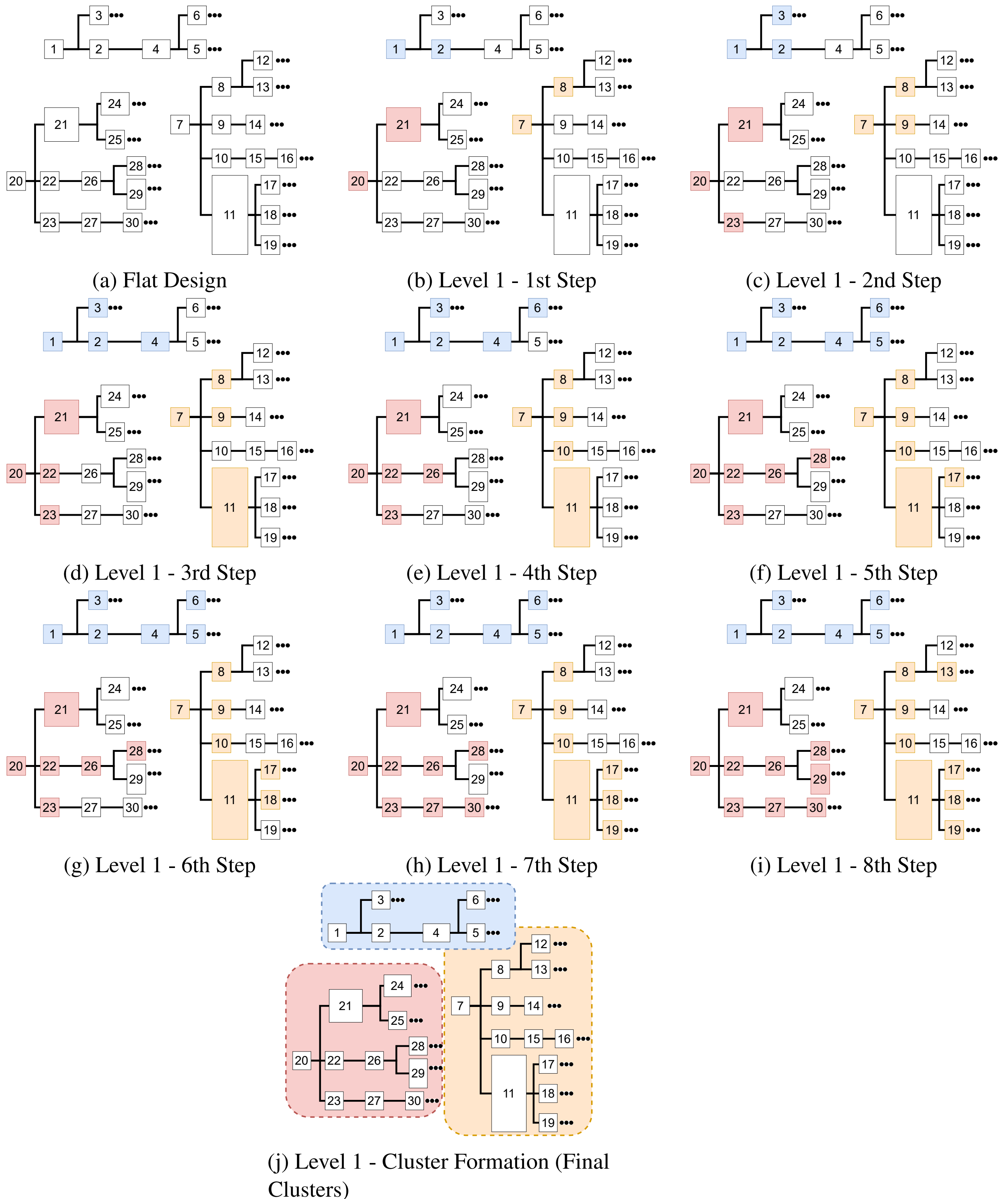
At this point, we introduce the multi-level Hybrid Hyper-Edge Coarsening (HHEC) algorithm that is able to grow a cluster not only based on the starting net (as Breadth Hyper-Edge Coarsening), but also based on the transitive nets. In this way, the clusters are not restricted to a net so they can grow at the same level exploiting more options. To describe the HHEC algorithm, the same example as BHEC is used (Figure 5.1a).

Similar to the BHEC algorithm, HHEC traverses the nets and assigns each cell to a cluster, while all clusters' area must satisfy the specified area ratio. So, based on this example, HHEC creates the blue, red and orange clusters by adding *Cells 1, 2*, *Cells 20, 21*, and *Cells 7, 8* respectively. Again, despite the fact that *Cell 11* drives a large fanout, the algorithm does not take this cell at the fist step into account, as it violates the clusters area ratio. Thus, the first two steps will be the same as the BHEC algorithm, but as far as the 3rd step is concerned, the algorithm inserts *Cell 4* in the blue cluster as this cell is in the transitive fanout of the net driven by *Cell 1*. This option, gives the algorithm the opportunity to grow the clusters much faster and also explore alternative solutions.

In this case, the blue cluster's area is 4.5, allowing *Cell 11* to be assigned to the orange cluster (area: 11), as in this step the upper area bound is $2.5 * 4.5 = 11.25$. The algorithm continues till the next steps by adding one cell each time to each cluster. The final clusters are depicted in Figure 4.7j, which are identical to the clusters that have been created by the BHEC algorithm, whereas the HHEC algorithm creates the clusters in a single level.

It is worth mentioning that in large and more complicated circuits the two algorithms result in extremely different solutions, not only in terms of the number of levels, but also in terms of clusters. However, both of them guarantee the area ratio of the clusters, which is very important for the physical design flow, and specifically for the placement step.

Institutional Repository - Library & Information Centre - University of Thessaly
30/12/2021 10:19:35 EET - 137.108.70.14



(a) Flat Design (b) Level 1 - 1st Step (c) Level 1 - 2nd Step

(d) Level 1 - 3rd Step (e) Level 1 - 4th Step (f) Level 1 - 5th Step

(g) Level 1 - 6th Step (h) Level 1 - 7th Step (i) Level 1 - 8th Step

(j) Level 1 - Cluster Formation (Final Clusters)

Figure 4.7 Hybrid Hyper-Edge Coarsening Algorithm Example

Institutional Repository - Library & Information Centre - University of Thessaly
30/12/2021 10:19:35 EET - 137.108.70.14

4.3.3 Clustering Summary

Clustering is very useful to automatically identify groups of standard cells that have stronger connectivity than other standard cells. For the tier assignment problem, clustering can be used as a divide-and-conquer technique to reduce the complexity of the partitioning problem.

The quality of the clusters must be high, because it dramatically influences the quality of the partitioning results. As a consequence, the clustering algorithm must take several parameters, such as internal and external connectivity, the area of the clusters, and also circuit delay into account.

Table 4.3 Clustering Approaches and Nets Fanout Order Options Comparison

Clustering Approach \ Fanout Order	Ascending	Descending
BHEC	Vertical Connections Reduction	Connectivity Density Reduction High Fanout Nets Reduction
HHEC	Vertical Connections Reduction Timing Paths Clustering Objects Reduction Per Level Less Runtime	Connectivity Density Reduction High Fanout Nets Reduction Timing Paths Clustering Objects Reduction Per Level Less Runtime

Table 4.3 summarizes the key results of our two examined clustering algorithms, the Breadth and Hybrid Hyper-Edge Coarsening algorithms (BHEC and HHEC respectively) based on the ascending and descending net fanout ordering. Due to the fact that HHEC can grow the clusters in a hybrid way (in breadth and depth), it is able to group entirely all the cells of a critical path, specifically when timing information is used as an important factor. Additionally, it can result in fewer levels of clustering, and so it can create clusters in less execution time.

On the other hand, the difference between ascending and descending orders is the purpose of the clustering use. If the target is to use few vertical interconnections then it is better to use the ascending order, whereas if the number of vertical interconnection is not the main goal, as for monolithic 3D ICs, the descending order can be used to reduce interconnecting density and also high fanout nets.

Institutional Repository - Library & Information Centre - University of Thessaly
30/12/2021 10:19:35 EET - 137.108.70.14

Chapter 5

Fine-Grained 3D IC Tier Assignment

3D integration can be applied either to a finer or to a higher level of circuit representation. In the end, each standard cell must be assigned to a tier by satisfying several parameters, such as the number of inter-tier connections, the wirelength, and the timing delay of the design. As mentioned in previous chapters, tier assignment is an important step in 3D integration, as it directly influences the placement solution and as a result the final Power, Performance, and Area (PPA).

Compared to the Block-Based methodologies that are described in the previous chapter, we present two additional layer assignment options that function on the flat netlist, *i.e.* the Flat Bin-Based Partitioning and the first 3D legalizer. We classify these methodologies as Fine-Grained since both operate on standard cells directly. Flat bin-based partitioning enables the use of traditional partitioning algorithms, such as Fiducia-Mattaessees (FM), effectively on very large designs by taking the physical positions of standard cells after the placement process into consideration. Similarly, the 3D legalizer starts from a placement with overlapping cells, and it assigns each cell to a tier taking for example the vertical interconnections, the impact on the initial solution, and the total wirelength into account. In the remaining part of this chapter the flat bin-based Partitioning and 3D legalization are presented in greater detail.

5.1 Partitioning

Flat partitioning is a physical approach for the layer assignment problem, while it is commonly used in the literature for 3D integration [66–69, 69, 70]. The partitioning goal is to reduce the connections (cutsize) between the partitions, while in 3D integration, this is interpreted as reducing the number of vertical interconnects. However, the problem of partitioning algorithms is that there is no easy way to force partitioning to achieve a pre-defined number

Institutional Repository - Library & Information Centre - University of Thessaly
30/12/2021 10:19:35 EET - 137.108.70.14

of vertical interconnect. Nevertheless, partitioning remains a very important approach as it can guarantee area constraints and minimum vertical interconnects.

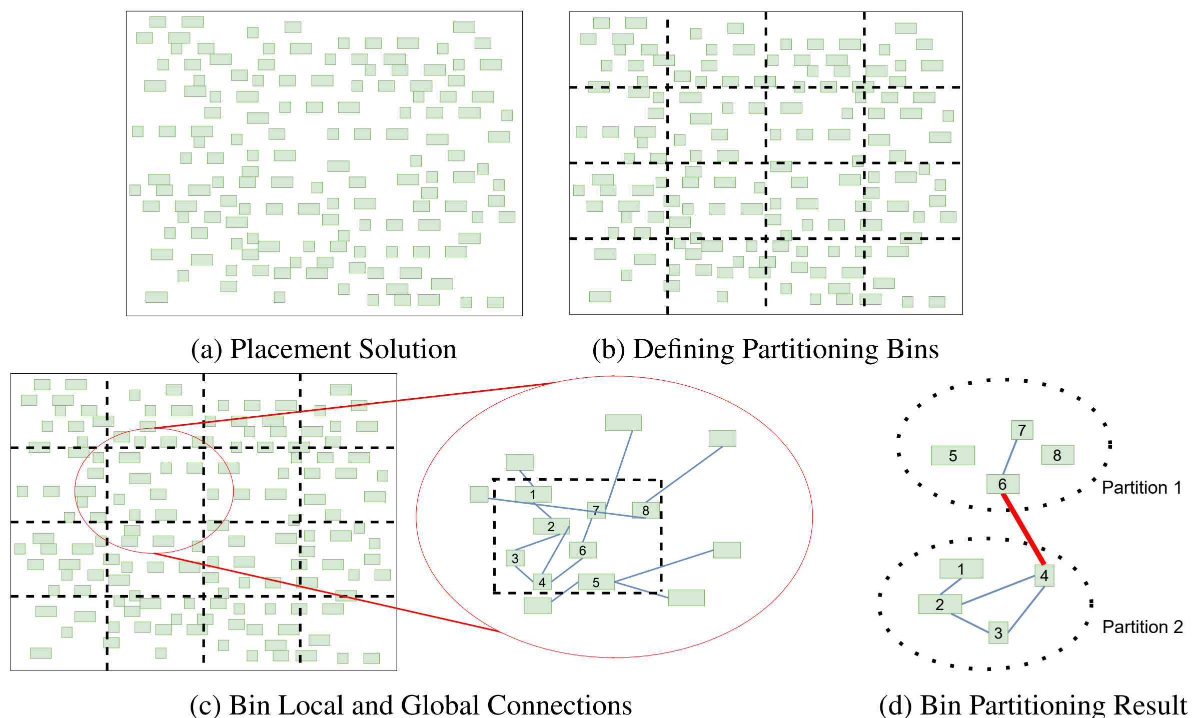
5.1.1 Bin-Based FM Partitioning

Modern circuits contain billions of cells, while they lead flat partitioning algorithms, such as FM, to extreme high execution time. However, there are several ways to reduce the runtime of the partitioning problem, besides using divide-and-conquer techniques as clustering (see Section 4.3). We present a placement-aware algorithm to partition a flat design not only without high runtime, but also with insignificant accuracy loss.

The Flat Bin-based FM (FBBFM) approach is a placement-aware partitioning scheme based on [66]. There are two options for bin-based FM. The first option, is a completely local FM, per bin, with bin cutsize being the cost function, where nets that span beyond the bin are ignored. This is a greedy approach assuming a fixed bin size, while it achieves O(C) complexity [66] with C being the total number of design components. This complexity also stems from the fact that the number of nets per bin are fixed. Another option is the global cutsize FM, where FM is performed on a bin by bin basis, while all nets are taken into account. This approach is not greedy, as by considering all nets, the global cutsize, and not the bin cutsize, is minimized. Thus, global cutsize FM has a global view, but it is of higher computational complexity. In this work we decided to implement global cutsize FM, to assess its scalability, cutsize, and 3D Quality of Results (QoR) and compared it to other Fine-Grained approaches, such as 3D legalization.

Figure 5.1 illustrates how the FBBFM approach works. Starting from a placement solution (either legal or illegal), depicted in Figure 5.1a, the chip area is divided into bins (Figure 4.6b), while the number and the area of the bins may differ for each design. We have noticed that bin sizes affect cutsize, as in [58], and execution time significantly. If the bin size is small, cutsize will be large, yet execution time will be reduced. On the other hand, as bin size increases, cutsizes tend to decrease, yet execution time also increases. Then, bins are partitioned sequentially (Figure 4.6c) by utilizing FM algorithm (Figure 4.6d). FM cutsize is computed incrementally, per move, and once a cell is assigned to a tier, it becomes locked and cannot move between tiers again. Once all cells of the particular bin have been examined, partitioning will be performed to the next bin, while the partitioning area balance factor is per bin. Due to the fact that the FM cutsize view is global, cutsize monotonically decreases, as bins are partitioned.

Institutional Repository - Library & Information Centre - University of Thessaly
30/12/2021 10:19:35 EET - 137.108.70.14



(a) Placement Solution (b) Defining Partitioning Bins

(c) Bin Local and Global Connections (d) Bin Partitioning Result

Figure 5.1 Flat Bin-Based FM (FBBFM) Approach Example

5.2 3D Legalization

For the 2D physical design flow, 3D flow must take several constraints into consideration. Power, performance, area, routing congestion, and thermal hot-spots are few constraints that are important when designing a modern circuit. As a consequence, the approaches that are used to deign a 3D circuit must be able to take all these constraints into account. Legalization is a very flexible step, where all these constraints can be applied. As a result, we convert a 2D legalization algorithm to perform legal moves into three dimensions, by taking total wirelength, placement density, and the number of vertical interconnections [11] into account. The legalizer can also be easily modified to support other important parameters for 3D integration, such as thermal hot-spots, etc.

3D legalization is essentially a form of partitioning, but it follows a sequential process. The algorithm legalizes, and simultaneously partitions cells across multiple tiers, by identifying the minimum displacement, or minimum wirelength position, or a combination of the two, considering all tiers, per cell. One of the advantage of this approach is that it is not limited to two tiers. Tier to tier spacing and tier index are used to incorporate z-distance into cell displacement and net 3D wirelength. We use the Abax [11] algorithm for 3D legalization, but we modified it to meet a user defined maximum number of vertical interconnections.

Institutional Repository - Library & Information Centre - University of Thessaly
30/12/2021 10:19:35 EET - 137.108.70.14

5.2.1 Unconstrained 3D Legalizer

Unconstrained 3D legalization [11] focuses on achieving optimal 3D wirelength reduction. Thus, the results of this approach may effectively serve as an upper bound for 3D gains, and also produce a required cutsize, *i.e.* number of vertical interconnection, to achieve these gains. However, the final legal placement may contain many vertical interconnection. Consequently, the unconstrained 3D legalizer can be used when the number of vertical interconnections is not a target for the 3D integration, as in monolithic 3D ICs.

Algorithm 1: Unconstrained 3D Legalization

```
1  Input: Netlist Cells (N), Tiers (T)
2  Output: Legal Placement with Minimum Cost
3  best_cost = ∞;
4  i = 1;
5  for each cell c_i in N do
6      for each tier T_i in T do
7          Legalize c_i on tier T_i;
8          3D_cost = compute_3D_cost();
9          if (3D_cost < best_cost) then
10             Update best_cost and best solution;
11     end
12 end
```

Algorithm 1 presents the top level pseudocode of the unconstrained 3D legaliser [11]. The inputs of the algorithm are a set of cells (N) and a tiers' structure (T). N contains information about the cells, *i.e.* their size, their coordinates, etc., while T contains information about the number, the dimensions, the placement rows, etc. of the tiers. The output of the algorithm is a legal placement with minimum cost. The cost can be the wirelength, the displacement, the delay, etc.

Each cell c_i is assigned and legalized on each tier T_i (lines: 5-7), while the cost for each assignment is computed (line 8). The legal position, *i.e.* x,y-coordinates and the tier, with the smallest cost will finally be the position of the cell. The vertical distance of the tiers is also included in the computation of the cost.

To legalize cell c_i in tier T, the core area of the latter is divided into rows and sub-rows. Rows are actually the placement rows, where all the cells must be aligned, while the sub-rows are parts of the rows when obstacles (*e.g. memory blocks*) exist. Algorithm 2 presents the high-level pseudocode of cell legalization to a tier. This pseudocode is the 2D algorithm proposed in [11], however the computation of the legalization cost includes the vertical interconnections wirelength. More specifically, each sub-row, of each row, is examined (lines:

Institutional Repository - Library & Information Centre - University of Thessaly
30/12/2021 10:19:35 EET - 137.108.70.14

5-15) to find the best position to legalize the examined cell (line: 16). In the case where the cell does not exceed the target density of the sub-row (line: 7), the cell is temporarily legalized to compute the 3D cost (lines: 8-13).

Algorithm 2: Legalize Cell To Tier

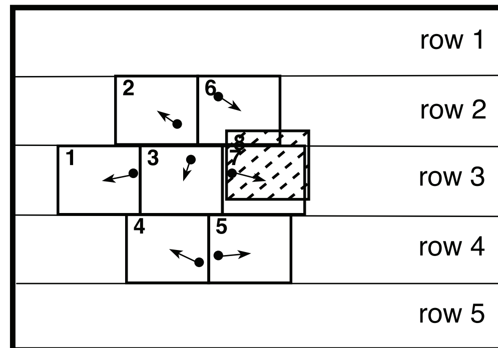
```
1  Input: Cell (C_i), Tier (T)
2  Output: Legal Position
3  B = # placement rows(T);
4  best_cost = ∞;
5  for each row r_i in B do
6      for each sub-row S_R of r_i do
7          if cell_fits_in_subrow(C_i, S_R) then
8              Legalize C_i in S_R;
9              D = compute_3D_cost();
10             if D < best_cost then
11                 Update best legalization;  /* tentative */
12                 best_cost = D;
13             B = min(cost_to_row_range(best_cost), B);
14     end
15 end
16 Select best legalization;
```

Figure 5.2 presents a simple example of cells' legalization. Lets assume that there are seven already legalized cells, *i.e.* Cells 1-7, while Cell 8 is the examined cell. Figure 5.2a shows the locations where already legalized cells have been placed, while arrows present their displacement from their initial positions. The dashed box represents the projection of Cell 8 to the tier. As previously mentioned, the examined cell will be tentatively legalized to the rows, while it will be legalized to the row where the cost will be minimum. As described in [11], the cost includes either the cost of the examined cell or the cost of all influenced cells. For example, in Figure 5.2b the total cost of the influenced cells, *i.e.* Cells 1, 3, and 7, and the cost of the examined Cell 8 is greater than the total cost of legalizing Cell 8 to the second row. So, finally, Cell 8 will be placed to the second row.

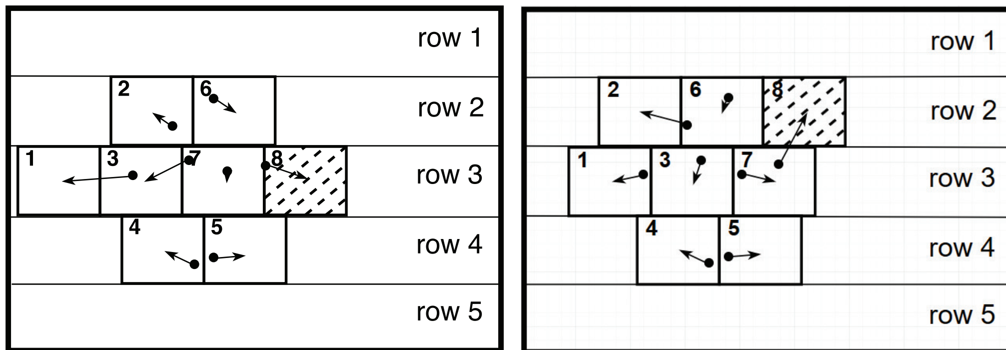
5.2.2 Constrained 3D Legalizer

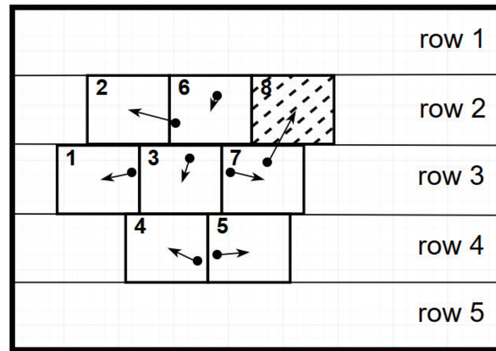
The 3D legalization algorithm can be constrained [71], by imposing an upper bound for the maximum permitted number of vertical interconnection, V. The advantage of 3D legalization over bin-based FM, is that the former can guarantee that the bound is satisfied, albeit at the expense of TAR. Thus, by sweeping the maximum number of vertical interconnection constraint, a trade-off curve between the number of vertical interconnection and 3D gains

Institutional Repository - Library & Information Centre - University of Thessaly
30/12/2021 10:19:35 EET - 137.108.70.14



(a) Tier Placed Cells and Examined Cell 8





(b) Sub-optimal Legal Placement for Cell 8 (c) Optimal Legal Placement for Cell 8

Figure 5.2 Legalize Cell to Sub-row

may be obtained. Such a bound is highly relevant to Stacked Die 3D, where the number of Hybrid Bond pads is constrained, *e.g.* 1 Hybrid Bond pad every 1.4μm.

To guarantee the vertical interconnection bound, V, the sequential 3D legalization algorithm [11] is modified to perform a limited number of different tier assignment moves. These moves are spread uniformly across the legalization process. Algorithm 3 describes the modified constrained 3D legalization algorithm.

Before legalizing each cell to a tier, we examined if the upper vertical interconnection number is violated, by computing the new number of the vertical interconnections. If the new number is greater than the upper bound, the cell will be placed to its original tier. On the other hand, if vertical interconnections do not exceed the bound, the cell can be legalized on both tiers. Note that if there are more than two tiers, the vertical interconnection overhead is computed for each tier. A 3D legalization move tentatively legalizes the current cell on all available tiers T, taking 3D displacement or 3D wirelength cost, or a combination of the two into consideration, and performs the minimum cost legalization move, lines 9-12. Thus, a cell will end up on a different tier to T_0, only if lower cost is achieved. Default,

Institutional Repository - Library & Information Centre - University of Thessaly
30/12/2021 10:19:35 EET - 137.108.70.14

Algorithm 3: Constrained 3D Legalization

```
1  Input: Netlist Cells (N), vertical interconnection Bound (V), Tiers (T)
2  Output: Legal Placement with Minimum Cost
3  best_cost = ∞;
4  i = 1;
5  for each cell c_i in N do
6      cutsize = compute_vertical_interconnections(c_i);
7      if (cutsize < V) then
8          for each tier T_i in T do
9              Legalize c_i on tier T_i;
10             3D_cost = compute_3D_cost();
11             if (3D_cost < best_cost) then
12                 Update best_cost and best solution;
13         end
14     else
15         Legalize c_i on tier T_0;
16         2D_cost = compute_2D_cost();
17         Update best_cost and best solution;
18 end
```

2D legalization is performed otherwise, lines 15-17. In this way, we guarantee that only a maximum of V cells is moved into another tier, line 7.

By specifying different number of vertical interconnection constraints, it is feasible to use constrained 3D legalization to explore the solution space between the bin-based FM solution, *i.e.* cutsize lower bound, and the unconstrained 3D legalization, *i.e.* cutsize upper bound, and evaluate 3D QoR results. This evaluation is described in Chapter 6.

Institutional Repository - Library & Information Centre - University of Thessaly
30/12/2021 10:19:35 EET - 137.108.70.14

Institutional Repository - Library & Information Centre - University of Thessaly
30/12/2021 10:19:35 EET - 137.108.70.14

Chapter 6

Experimental Flows and Results

The first part of this chapter presents the experimental flows that have been used, while the second part demonstrates the experimental results of the examined approaches. Figure 6.1 summarizes the experiments for both Fine-Grained and Block-Based tier assignment approaches.

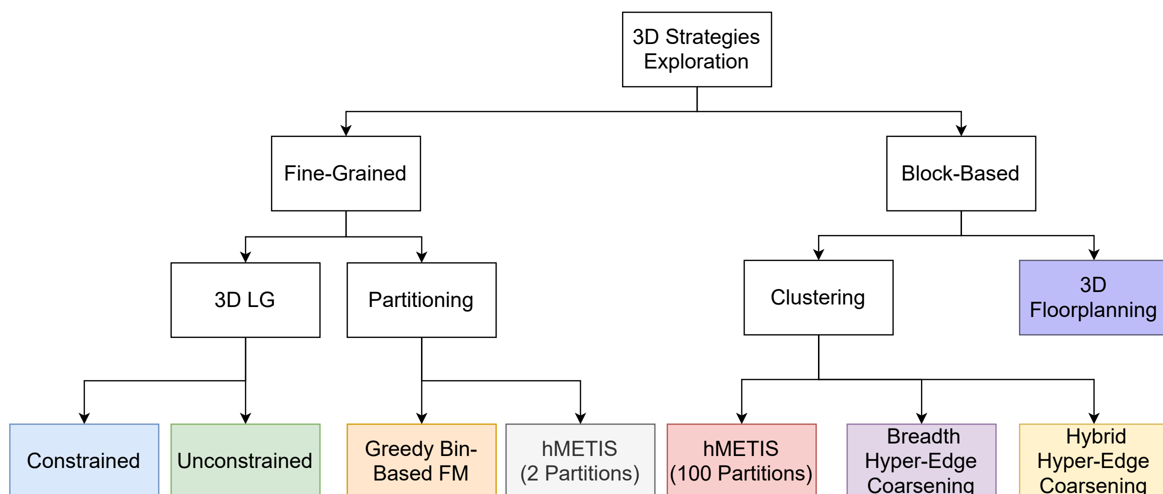


Figure 6.1 Examined Approaches Tree

6.1 Experimental Flow

Firstly, we explain the Fine-Grained methodologies flow, that includes the Flat Bin-Based FM partitioning, the unconstrained legalizer and the hMETIS Bi-partitioning. Secondly, we present the Block-Based tier assignment flows for our 3D floorplanning algorithm, the kMETIS tool, and our custom clustering tool.

Institutional Repository - Library & Information Centre - University of Thessaly
30/12/2021 10:19:35 EET - 137.108.70.14

6.1.1 Fine-Grained Tier Assignment Flow

In this section, we present the design flows used to compare the Fine-Grained approaches. More specifically, we compare the Greedy Bin-Based FM partitioning (GBBFM), the 3D Legalizer (3DLG), and the partitioning of hMETIS tool. We use the 2D benchmark circuits as baseline with the 2D circuits undergoing the exact same process for timing optimization.

Figure 6.2 Fine-Grained Tier Assignment Flow

Institutional Repository - Library & Information Centre - University of Thessaly
30/12/2021 10:19:35 EET - 137.108.70.14

Figure 6.2 shows the steps of our 3D IC flow in detail. These consist of three main parts, the initialization step, where the 2D design is prepared for 3D tier partitioning, the tier assignment step where cells are assigned to the bottom or top tier, and the timing optimization step where the multiple tier netlist is PPA-optimized. We describe these steps in detail below.

Initialization Step

To perform a fair comparison for 2D and 3D circuits, the 3D flow starts from the same netlist as the 2D flow. A modified LEF file with shrunk cell dimensions is used. This LEF contains the same cells as in 2D flow, but the width and height of the cells are modified to be equal to the minimum acceptable placement dimensions, *i.e.* the placement site width and height, respectively. This modification is necessary to enable the conventional industrial P&R tool to place components into half of the original 2D footprint area, otherwise utilization would be greater than 100%. Thus, by using the shrunk cells' LEF, standard industrial P&R flow can be applied to place the standard cells into half of the area that is used for the conventional 2D flow. After loading the netlist and the LEF file, we perform floorplanning, placement and In-Place Optimizations (IPO) into half of the original 2D area. For timing optimization, we use a multiple IPO strategy. Instead of performing a single IPO run, we perform three IPO iterations, as shown in Figure 6.3, aiming to determine and use the best overall IPO result. The reason why multiple IPOs are used, is because the IPO process is path based, while it exhibits randomness, meaning that fixing the timing of some critical paths, disturbs the timing of others, or even increases the cell area. In addition, the IPO algorithm sometimes terminates prematurely, resulting in an arbitrary solution, which may be worse than the solution prior to the IPO step. Having optimized the Shrunk 2D design, we extract the new timing optimized netlists, *i.e.* three of them, are written out as a DEF file for the next step, *i.e.* the tier partitioning step.

Tier Assignment Step

In this step, cells are assigned to tiers, using either the GBBFM, the 3DLG, or the hMETIS tier assignment approach. Cell sizes are also restored, *i.e.* the original LEF, not the shrunk size is used (as in the initialization step). Each partitioning approach optimizes different metrics. GBBFM minimizes grid cutsize, *i.e.* number of vertical interconnections, by assigning strongly connected components of the same grid to the same tier. Similarly, hMETIS minimizes the cutsize of the entire circuit, by using multi-level clustering. On the other hand, 3DLG minimizes components displacement from their original placement position. In our flow, the initial positions of the components, for the assignment step (*i.e.* the

Institutional Repository - Library & Information Centre - University of Thessaly
30/12/2021 10:19:35 EET - 137.108.70.14

partitioning), correspond to the post IPO positions. In this way, cells are legalized as close as possible to their original positions and the 3DLG assigns them into the tier, where the minimum perturbation occurs.

Optimization Step

After assigning cells to tiers, timing-driven optimization is performed to each tier, by taking the tier assignment changes into consideration. Thus, the optimized 2D netlist and the shrunk cell LEF are utilized, as in the initialization step. Cell tier assignment is used to perform IPO on a tier by tier basis, *i.e.* keeping other tier cells soft fixed (SOFTFIX attribute). This allows the cells of one tier to be fully optimized, while the cells of the other tier are allowed to only be upsized or downsized. Thus, there is no need to add any virtual I/O pins for tier by tier optimization, as [72] proposes. We have concluded, based on our experiments that three IPOs are sufficient and further IPO runs do not make significant QoR differences. Figure 6.3 shows the multi-stage IPO process for the 3D designs in detail.

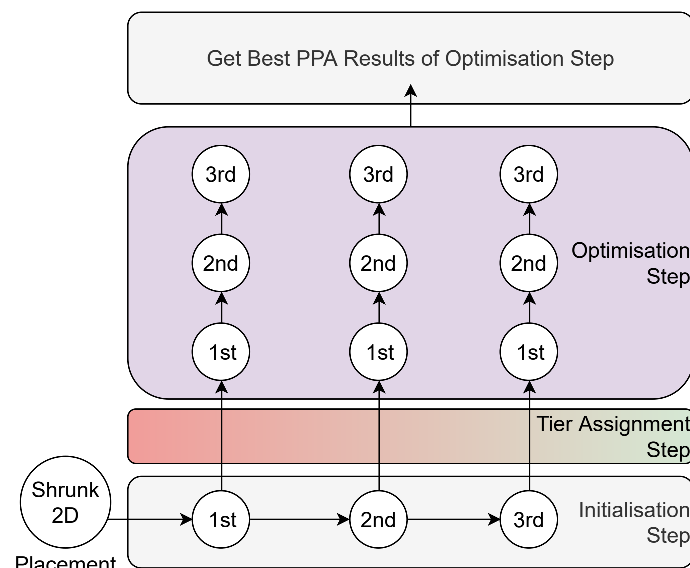


Figure 6.3 3D Multiple IPOs Flow

6.1.2 Block-Based Tier Assignment Flow

Similarly to the Fine-Grained, our Block-Based methodologies flows contain several steps, shown in Figure 6.4. At the first step, *i.e.* the initialization, the appropriate files are loaded, while at the second step three different approaches are compared. The first floorplans the

Institutional Repository - Library & Information Centre - University of Thessaly
30/12/2021 10:19:35 EET - 137.108.70.14

Table 6.1 Single vs. Multiple IPO Impact to 3D Flows - Best Single/Multiple IPO Results

Benchmark	Partitioning Methodology	IPO	Power (mW)	Circuit delay (ns)	Violating Paths	Total Components Area (um^2)
ldpc	Greedy BBFM	Single	1064.763	2.0165	2046	267206.082
#Cells = 96361		Multiple	797.023	1.6275	0	266638.970
#Nets = 98412	3DLG	Single	798.605	1.6595	367	267206.082
#IOs = 4100		Multiple	790.288	1.6275	0	266874.670
fft	Greedy BBFM	Single	630.063	1.481	16	614178.838
#Cells = 194175		Multiple	624.466	1.395	0	613528.728
#Nets = 194219	3DLG	Single	638.795	1.395	0	615576.253
#IOs = 90		Multiple	634.988	1.395	0	615426.188
jpeg	Greedy BBFM	Single	2258.869	1.5264	53	1730458.782
#Cells = 576366		Multiple	2257.626	1.4694	0	1730142.602
#Nets = 576394	3DLG	Single	2249.683	1.4954	46	1735985.287
#IOs = 67		Multiple	2244.926	1.4744	4	1735112.334

physical hierarchy modules of the netlist, the second utilizes the METIS tool, while the third approach utilizes our clustering algorithm.

The next step is to determine the tier of each block. So, for the floorplanning flow, we use our 3D floorplanning algorithm that is based on the circuit functional blocks, assigning each block to a tier by aiming to create two equal size tiers. While, for the clustering approaches, *i.e.* kMETIS and our clustering tool, each block is assigned to a tier, by using our 3D legalisation algorithm. Finally, the optimization step takes the results of the previous step into consideration (either the 3D floorplanning or the clustering step), and performs placement and IPOs. The main steps are described with more details below.

3D Floorplanning Step

3D floorplanning is based on circuit architecture, while the netlist modules are assigned to tiers using the approach proposed in Section 4.1. The floorplanning approach is better applicable when the design is modular, *i.e* each module will be well-separated from the others and their connectivity will contain only the appropriate connections. In this way, there will be few connections between the modules, resulting in few vertical interconnections for 3D integration. However, since the floorplanning approach is strongly correlated with the circuit's architecture, which is coming from the experience of the designers, there are designs that may contain an appropriate number of modules, while there are others that do not contain well-defined modules hierarchy. Designs without well-defined modules present less or no gains after 3D integration. In this way, to evaluate this approach, we first extract the design modules, we perform 3D floorplanning utilizing the algorithm proposed in Section 4.1, and

Institutional Repository - Library & Information Centre - University of Thessaly
30/12/2021 10:19:35 EET - 137.108.70.14

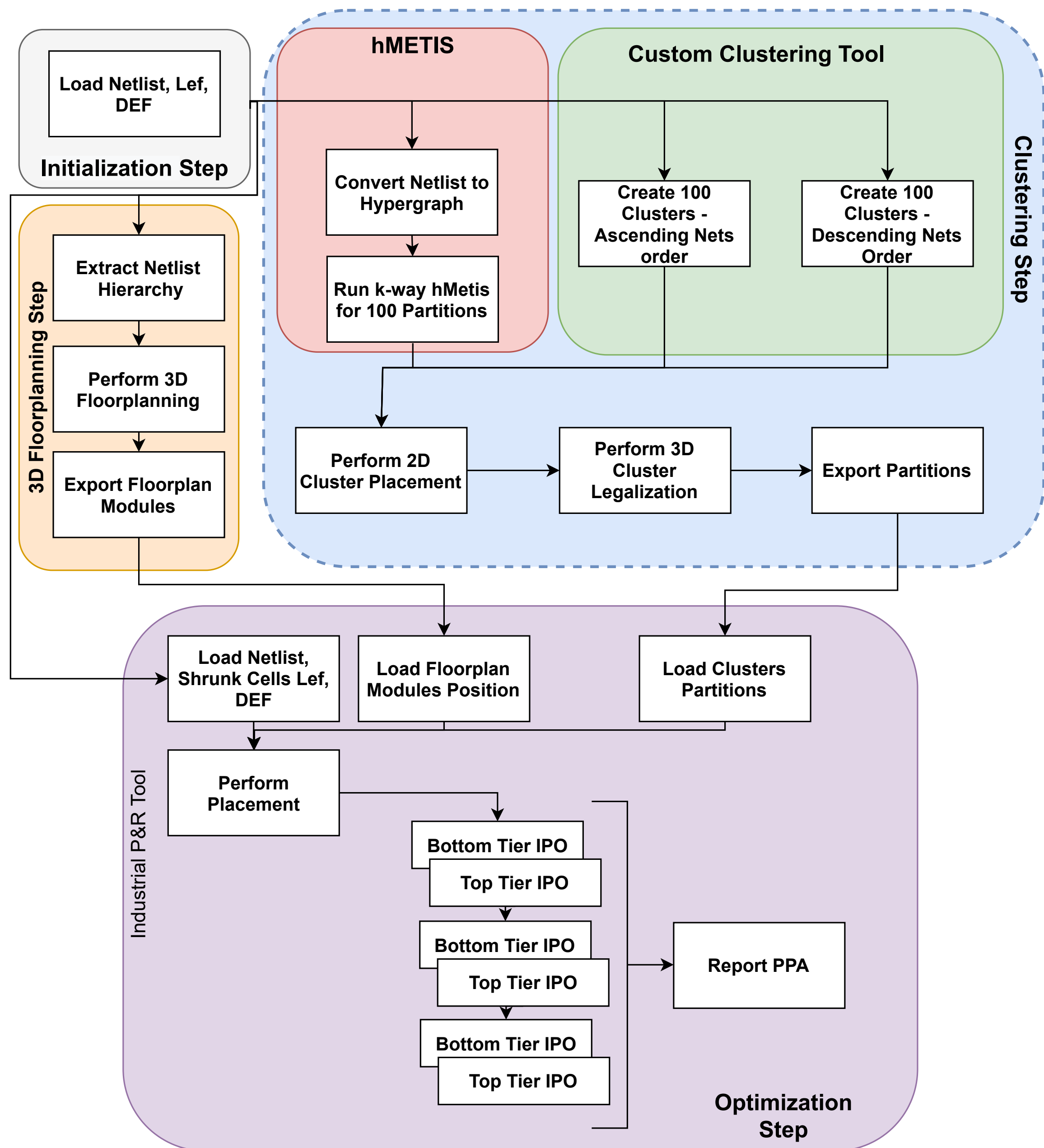


Figure 6.4 Block-Based Experimental Flow

then we export the positions and tier assignment of the modules to be used as constraints in the optimization step.

3D Clustering Step

Due to the fact that floorplanning depends on the functional modules of the netlist, we propose that clustering may lead to better PPA results for the 3D integration. Clustering

Institutional Repository - Library & Information Centre - University of Thessaly
30/12/2021 10:19:35 EET - 137.108.70.14

approaches are able to identify groups of cells that have strong connectivity, similarly to a well designed modular circuit, but without human interference.

To evaluate clustering, we compare the state-of-the-art METIS clustering tool [36] and our custom clustering tool (Section 4.3.2). For the METIS tool, we convert the input netlist into a hypergraph, as this is the required format of the tool, and then we execute the k-way flavor to create the clusters. It is important to note that METIS is a closed source suite that contain several algorithms, so it is not feasible to utilize each tool separately, as the clustering algorithm. In several works, METIS is used as a clustering algorithm by defining each partition as a cluster. As a consequence, in our experiments 100 clusters have been created by requesting 100 partitions from the tool.

On the other hand, our clustering approach directly creates the required number of clusters, *i.e.* 100 for the current experiments. First, we sort the nets in descending or ascending fanout order and then we utilize either the Breadth Hyper-Edge Coarsening Algorithm or the Hybrid Hyper-Edge Coarsening Algorithm, which have been described in Section 4.3. After clusters' formulation, we utilize our 3D legalization tool to find the best tier and position for each cluster, by taking the number of inter-connections, and the tiers area ratio and the minimum perturbation of the 2D clusters placement into account, as well. Finally, we export the clusters positions and their tier assignment to be used as constraints in the optimization step.

Optimization Step

This step is similar to the optimization step of the Fine-Grained flow, but instead of loading just the tier assignment to the industrial P&R tool, we define the regions of the modules or clusters from the previous step. So, in this step, we load the netlist and the shrunk cells' library, and the positions of the modules or clusters, as well as their tier. However, the standard cells are not placed yet, so we place the modules or clusters by taking their bounding boxes into account. So, each cell will be placed inside the bounding boxes of the module or cluster which it belongs to. Finally, we perform multiple IPOs for better PPA results.

6.2 3D Legalization Methods

In this section we compare the two flavors of our 3D legalizer, *i.e.* the constrained and unconstrained.

To compare the two 3D Legalization flavors, we used the 45nm nanoCAS library [73], which is a 3D variation of the 45nm NANGATE library, featuring explicitly defined top and bottom tier cells. This feature allows us to keep the cells of a specific tier fixed, while timing-driven optimization is performed for the other tier by using the industrial P&R tool.

Institutional Repository - Library & Information Centre - University of Thessaly
30/12/2021 10:19:35 EET - 137.108.70.14

In terms of wire delay, for the specific 45nm library, 1 Fanout-of-4 (FO4) inverter gate delay is proportional to 300μm RC wire delay. Thus, RC delay is not as significant for the library we used, as in more advanced technology nodes, *i.e.* 20*n*m and below, where the designs are wire-delay dominated and not gate-delay dominated, as in our case.

We selected four OpenCores benchmarks [74] for the comparison, and we utilize the Cadence Innovus tool for physical design. 2D and 2D shrunk cells placement is thus performed using Innovus, the latter performed by shrinking all cells to single site width. After 2D placement, we utilize our custom tool, and after tier partitioning, we move the design back to Cadence Innovus for timing-driven IPO.

Table 6.2 3D Legalization: Inter-tier Vias vs. 3D Wirelength Ratio

Benchmark	Cutsize	Cutsize Constraint					
		100%	90%	60%	40%	30%	20%
		3D Wirelength Ratio					
pid	3361	**1.00**	1.50	1.82	1.79	1.77	1.69
aes	5755	**1.00**	1.20	1.59	1.62	1.69	1.70
aes128	7334	**1.00**	1.55	1.98	1.99	2.56	1.87
ldpc	51599	**1.00**	1.00	1.96	2.08	2.11	2.07

Table 6.2 reports the inter-tier vias constraint to relative wirelength trade-off, for 3D legalization, where the 100% value corresponds to unconstrained 3D legalization. The latter achieves the smallest wirelength, while imposed cutsize constraints increase wirelength. Stricter constraints lead to higher wirelength results, *e.g.* `ldpc` benchmark's wirelength doubles, for a 40% cutsize constraint (60% cutsize reduction), compared to the unconstrained 3D legalization result. However, this trend is not monotonic, particularly for small inter-tier vias constraint value, *e.g.* 20%. At this point, wirelength may be improved, *e.g* benchmarks `aes128` and `ldpc`, as more and more standard cells are assigned to a single tier.

Table 6.3 presents comparative results, post the 3D tier assignment and timing driven IPO for the constrained and unconstrained 3D legalizer, compared to the 2D design. Parameters which are compared include (1) slack, (2) number of inter-tier vias (cutsize), (3) total routed WireLength (WL), (4) Tier Area Ratio (TAR) and (5) execution time. For the unconstrained 3D legalizer, we used three cutsize constraints, *i.e.* 90%, 60% and 30%, relative to the cutsize value achieved by the unconstrained 3D legalizer. Slack values in the table do not follow a uniform trend, for the benchmarks selected and 45nm library used. This is caused, as IPO operations significantly change the design slack profile. Further on, we do not observe a 1-1 correspondence between total WL and slack.

We observe, that despite the fact that the core area is reduced to half, the total wirelength and the circuit delay are not reduced significantly. This can be explained due to the higher

Institutional Repository - Library & Information Centre - University of Thessaly
30/12/2021 10:19:35 EET - 137.108.70.14

routing congestion and due to the fact that the examined benchmarks are gate dominated. The two flavors of the 3D legalizer, seem to have similar behavior in terms of total wirelength and slack, while the unconstrained flavor, results in more balanced tiers without significant increment on the number of vertical interconnections. Consequently, it is clear that if the number of vias is predefined, the constrained 3D legalizer will result in more unbalanced core areas, as presented in Figure 6.3. As a result, for the next comparisons we use the unconstrained flavor to compare the 3D legalizer to the other approaches.

Table 6.3 3D Legalization Schemes Comparison

Bench.	**Constrained 3D-Legalizer**					**Unconstrained 3D-Legalizer**					**2D**	
	Total WL (um)	**Inter-tier Vias Num.**	**Slack Ratio**	**Pre-IPO Tier Area Ratio**	**Exec. Time (s)**	**Total WL (um)**	**Inter-tier Vias Num.**	**Slack Ratio**	**Pre-IPO Tier Area Ratio**	**Exec. Time (s)**	**Total WL (um)**	**Slack Ratio**
pid	7.23E+04	3000 (90%)	1.002	2.385	0.18	6.80E+04	3361	1.005	1.308	0.24	7.10E+04	1.00
	7.38E+04	2000 (60%)	1.052	5.647	0.31							
	6.77E+04	1000 (30%)	1.003	12.943	0.26							
aes	1.87E+05	5000 (90%)	1.101	2.066	0.54	1.89E+05	5755	1.143	1.276	0.54	1.57E+05	1.00
	2.07E+05	3500 (60%)	1.017	5.151	0.72							
	2.16E+05	1800 (30%)	1.045	12.186	0.80							
aes128	1.95E+05	6600 (90%)	1.041	2.138	1.78	1.45E+05	7334	1.061	1.455	1.46	1.90E+05	1.00
	2.05E+05	4400 (60%)	0.867	5.157	2.93							
	2.02E+05	2200 (30%)	0.949	12.387	2.69							
ldpc	3.24E+06	46500 (90%)	1.091	2.062	32.94	3.17E+06	51599	1.049	1.153	47.92	3.41E+06	1.00
	3.13E+06	31000 (60%)	1.073	8.049	48.93							
	3.18E+06	15500 (30%)	1.055	12.061	39.90							
AVG	9.14E+05	-	**0.927**	-	-	8.92E+05	-	**1.065**	-	-	9.57E+05	**1.00**

6.3 Metal Stacking

3D integration offers two orders of magnitude, higher vertical interconnect density at a much lower footprint area compared to 2D. Therefore, the same number of pins as with a 2D circuit, must now be routed within half the wiring area. This situation considerably increases routing congestion, particularly, for the lower layers of the BEOL process. The severity of routing congestion is also a function of the interconnect metal stack (or technology) utilized in the manufacturing process.

To investigate the correlation between congestion and interconnect stack, we use the open-source 45nm nanoCAS library [75]. This is a 3D library, providing cells for two tier vertical integration. Figure 6.5 illustrates the library metal stack and routing layers. We call this metal stack Single, as the same metal layers are used for both device tiers.

The gray rectangles represent the individual metal layers, *e.g. m1*, *m2*, ..., *m12*, with the rectangle height drawn proportionally to the respective layer resistance, *i.e.* less resistive metals are taller. The blue cylinders illustrate layer to layer vias, with cylinder height again inversely proportional to via resistance. The red spots, indicate the layer where the cell pin

Institutional Repository - Library & Information Centre - University of Thessaly
30/12/2021 10:19:35 EET - 137.108.70.14

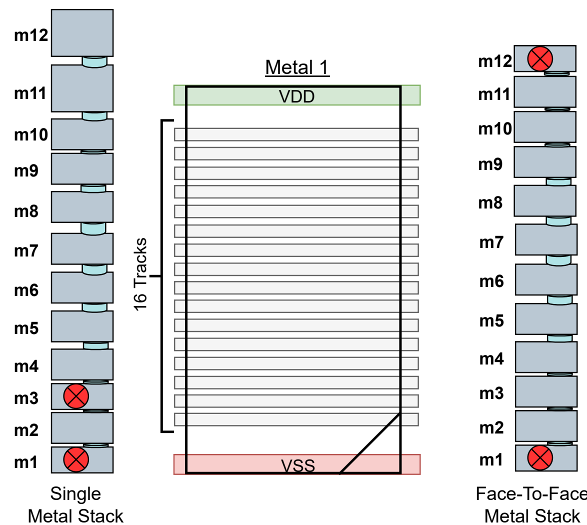


Figure 6.5 Metal Stacks and Tracks [8]

connects. Thus, in the 45 nm nanoCAS library [75], bottom tier cell pins are allocated to the first metal layer, while the top tier cell pins to the third metal layer. This layer pin assignment can lead to high routing congestion areas, as the two pin assignment metal layers are rather adjacent to each other. This assignment will force the router to over-utilize the lower metal layers as these are used both for intra- and inter-tier connections. To measure and compare the effect of congestion in the lower metal layers, we have created a new, alternative pin layer assignment for the 45 nm library, presented in Figure 6.5. Our alternative metal stack pin assignment contains the same number of layers but tiers are integrated Face-to-Face (F2F), meaning that bottom tier cell pins are at metal 1, top tier cell pin at metal 12. We also call this version as Double metal stack, as there are 6 metal layers for the bottom and 6 for the top tier.

6.3.1 Metal Stack Comparison

In this section we compare two different metal stacks, the Face-to-Face (Double) and the Single metal stack.

Figure 6.6 depicts congestion analysis, reported by the Cadence Innovus tool [14], of the *ldpc* OpenCores benchmark [74], for the 2D version and the two 3D metal stacks. We illustrate this benchmark as its high connectivity leads to higher routing congestion. As a result, 2D presents 0.07% over congested layers, while single metal stack presents 3.89% and F2F 0.56%, respectively. The low routing congestion of Double metal stack is caused because the same tier connections use their nearest metals, (*i.e.* the upper or lower), and do not interfere much with tier to tier connections. However, for the Double metal stack, routing

Institutional Repository - Library & Information Centre - University of Thessaly
30/12/2021 10:19:35 EET - 137.108.70.14

Layer	OverCon #Gcell (1)	OverCon #Gcell (3)	#Gcell OverCon
metal1	0(0.00%)	0(0.00%)	(0.00%)
metal2	2(0.00%)	0(0.00%)	(0.00%)
metal3	2(0.00%)	1(0.00%)	(0.01%)
metal4	0(0.00%)	0(0.00%)	(0.00%)
metal5	1(0.00%)	0(0.00%)	(0.00%)
metal6	2(0.00%)	0(0.00%)	(0.00%)
metal7	0(0.00%)	0(0.00%)	(0.00%)
metal8	1(0.00%)	0(0.00%)	(0.00%)
metal9	171(0.36%)	1(0.00%)	(0.36%)
metal10	99(0.21%)	0(0.00%)	(0.21%)
metal11	0(0.00%)	0(0.00%)	(0.00%)
metal12	0(0.00%)	0(0.00%)	(0.00%)
Total	278(0.07%)	2(0.00%)	(0.07%)

(a) 2D

Layer	OverCon #Gcell (1-2)	OverCon #Gcell (3-5)	OverCon #Gcell (6-8)	OverCon #Gcell (9-11)	#Gcell OverCon
metal1	12(1.06%)	0(0.00%)	0(0.00%)	0(0.00%)	(1.06%)
metal2	84(5.38%)	31(1.99%)	4(0.26%)	4(0.26%)	(7.88%)
metal3	369(34.2%)	67(6.21%)	0(0.00%)	0(0.00%)	(40.4%)
metal4	0(0.00%)	0(0.00%)	0(0.00%)	0(0.00%)	(0.00%)
metal5	0(0.00%)	0(0.00%)	0(0.00%)	0(0.00%)	(0.00%)
metal6	0(0.00%)	0(0.00%)	0(0.00%)	0(0.00%)	(0.00%)
metal7	0(0.00%)	0(0.00%)	0(0.00%)	0(0.00%)	(0.00%)
metal8	0(0.00%)	0(0.00%)	0(0.00%)	0(0.00%)	(0.00%)
metal9	0(0.00%)	0(0.00%)	0(0.00%)	0(0.00%)	(0.00%)
metal10	0(0.00%)	0(0.00%)	0(0.00%)	0(0.00%)	(0.00%)
metal11	0(0.00%)	0(0.00%)	0(0.00%)	0(0.00%)	(0.00%)
metal12	0(0.00%)	0(0.00%)	0(0.00%)	0(0.00%)	(0.00%)
Total	465(3.16%)	98(0.67%)	4(0.03%)	4(0.03%)	(3.89%)

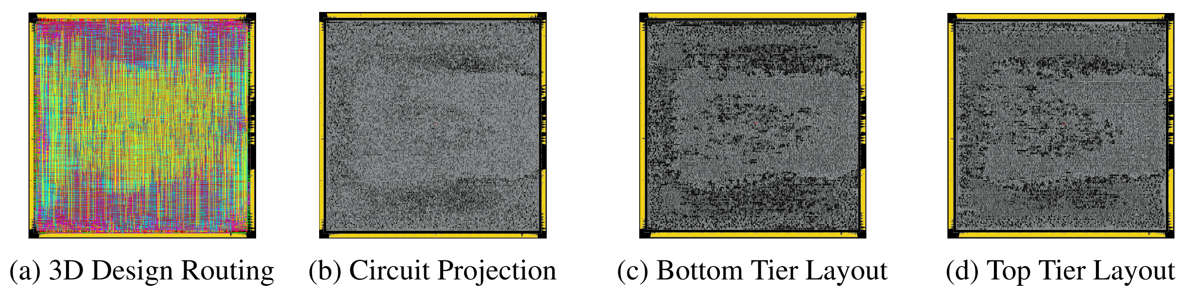
(b) Single Metal Stack

Layer	OverCon #Gcell (1-4)	OverCon #Gcell (5-8)	OverCon #Gcell (9-12)	OverCon #Gcell (13-16)	#Gcell OverCon
metal1	7(0.62%)	0(0.00%)	0(0.00%)	0(0.00%)	(0.62%)
metal2	52(3.33%)	14(0.90%)	2(0.13%)	2(0.13%)	(4.49%)
metal3	0(0.00%)	0(0.00%)	0(0.00%)	0(0.00%)	(0.00%)
metal4	0(0.00%)	0(0.00%)	0(0.00%)	0(0.00%)	(0.00%)
metal5	0(0.00%)	0(0.00%)	0(0.00%)	0(0.00%)	(0.00%)
metal6	0(0.00%)	0(0.00%)	0(0.00%)	0(0.00%)	(0.00%)
metal7	0(0.00%)	0(0.00%)	0(0.00%)	0(0.00%)	(0.00%)
metal8	0(0.00%)	0(0.00%)	0(0.00%)	0(0.00%)	(0.00%)
metal9	0(0.00%)	0(0.00%)	0(0.00%)	0(0.00%)	(0.00%)
metal10	0(0.00%)	0(0.00%)	0(0.00%)	0(0.00%)	(0.00%)
metal11	0(0.00%)	0(0.00%)	0(0.00%)	0(0.00%)	(0.00%)
metal12	25(1.84%)	0(0.00%)	0(0.00%)	0(0.00%)	(1.84%)
Total	84(0.46%)	14(0.08%)	2(0.01%)	2(0.01%)	(0.56%)

(c) F2F Double Metal Stack

Figure 6.6 ldpc Benchmark Congestion Analysis

interconnections become longer, leading to greater wirelength. Figure 6.7 shows the metal stack layout of the *ldpc* benchmark for a Single metal stack that is more congested.



(a) 3D Design Routing (b) Circuit Projection (c) Bottom Tier Layout (d) Top Tier Layout

Figure 6.7 *ldpc* Benchmark Routing and 3D Tiers Layout using Single Metal Stack

Institutional Repository - Library & Information Centre - University of Thessaly
30/12/2021 10:19:35 EET - 137.108.70.14

6.4 3D IC Methodologies Comparison

In this section we compare the experimental results of both Fine-Grained and Block-Based tier assignment techniques. Additionally, we present results for the two examined metal stacks, *i.e.* Single and Face-to-Face (Double) metal stacks, after the entire 3D P&R flow. We use 6 OpenCores benchmarks [74], *i.e. pid, aes, aes128, ldpc, fft,* and *jpeg*, to compare the examined approaches, while we used the Bin-Based FM partitioning approach results as point of reference, so all the results that are presented are ratios to the reference approach.

First, we compare the ratio between the number of needed vias and the number of nets, *i.e.* the vias per net. This metric is important as it shows two different characteristics. The first is the routed congestion and the second is the impact of the metal stack. So, in Figure 6.8 we can see that the Single metal stack needs more vias per net compared to the Double metal stack. This is caused since the Single metal stack (Figure 6.5) leads to greater congested placements, as mentioned in Section 6.3.1.

On the other hand, we can see that some approaches seem to be unaffected based on the selected metal stack. For example, the 3DFP (area balance floorplanning) and the kMETIS (use partitions as clusters) do now show significant improvement. This is caused as the vertical connections add many vias, *i.e.* 11 vias to connect a cell of the bottom tier to a cell of the top tier (see Figure 6.5). These two approaches present low congestion for the Double metal stack, but high number of 3D vertical interconnections. As a result, the vias per net is higher, which leads to higher resistance and impacts circuit timing.

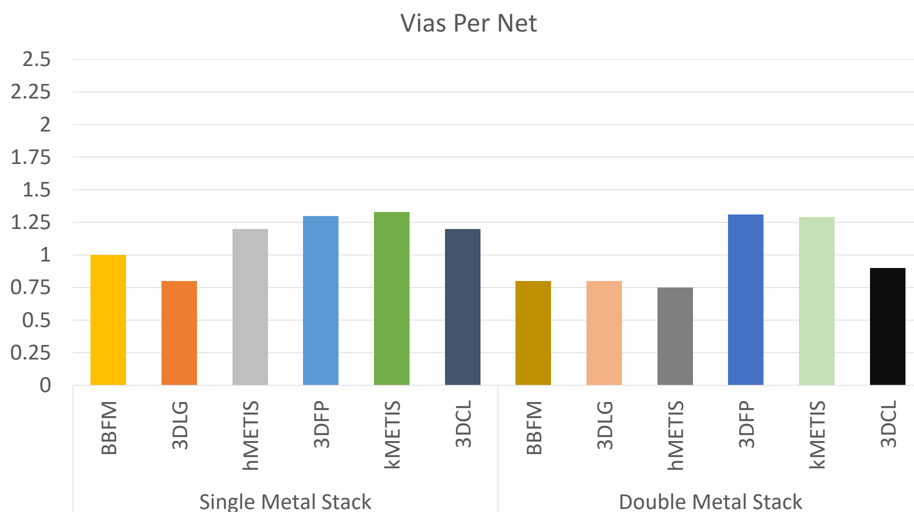


Figure 6.8 Vias Per Net Comparison Results

Figure 6.9 presents the congestion results of the examined approaches. This chart shows that the Double metal stack indeed leads to lower congested placement solutions.

Institutional Repository - Library & Information Centre - University of Thessaly
30/12/2021 10:19:35 EET - 137.108.70.14

Moreover, all the approaches present almost the same congestion, except for the 3DCL (our 3D clustering algorithm). As for the METIS tool approaches, *i.e.* hMETIS (partitioning) and kMETIS (create 100 partitions as clusters), higher congestion is presented, when Single metal stack is utilized. This occurs due to the fact that METIS creates partitions with few 3D vertical vias by using clusters. The clustered cells, that will be strongly connected, increase the tier (or local) density. As a result, the high cell density increase the routing congestion.

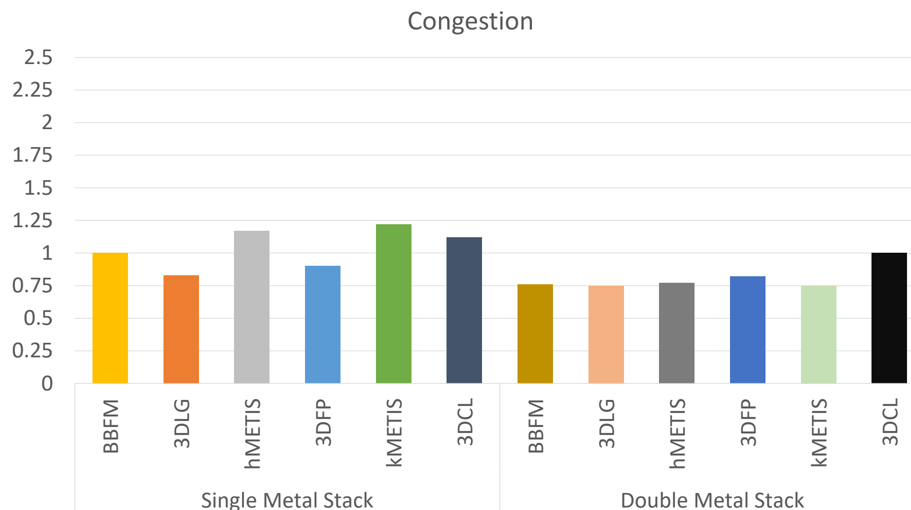


Figure 6.9 Congestion Comparison Results

For total power consumption, the examined metal stacks present similar results. Total power is the sum of Internal Power Switching Power, and Leakage Power. Figure 6.10 shows that hMETIS results in more power consumption than all the other approaches. This is caused since P&R tool adds many gates to meet the timing requirements, as can be seen in Figure 6.11, and so the internal power is increased.

One more factor that influences power, but also circuit delay and area, is the total wirelength. In Figure 6.12 we can see that 3DFP presents great wirelength both in Single and Double metal stack. This can be justified because floorplanning assigns large blocks of logic to a single tier, but the optimal solution may be to split the cells to different tiers, in order to reduce the length of large nets. Moreover, 3DCL presents the worst wirelength which is caused because the clustering algorithm groups the strongly connected standard cells all together, but since the vertical wires are longer compared to the Double metal stack, the total wirelength will be increased. On the other hand, 3DLG results in less wirelength when Single metal stack is used. It is important to note that the flavor that it used for the 3DLG is the unconstrained, *i.e.* there is no vertical vias constrained, while the target is to minimize the perturbation of the initial placement with overlaps.

Institutional Repository - Library & Information Centre - University of Thessaly
30/12/2021 10:19:35 EET - 137.108.70.14

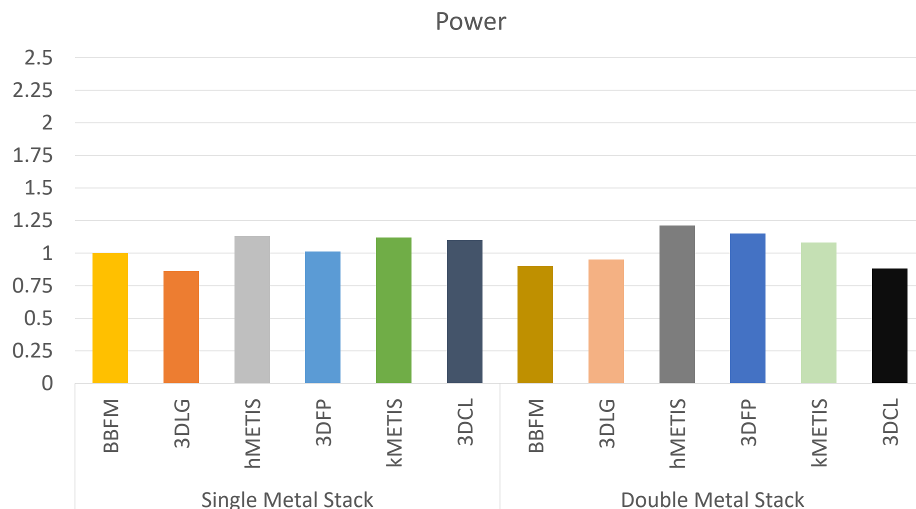


Figure 6.10 Power Comparison Results

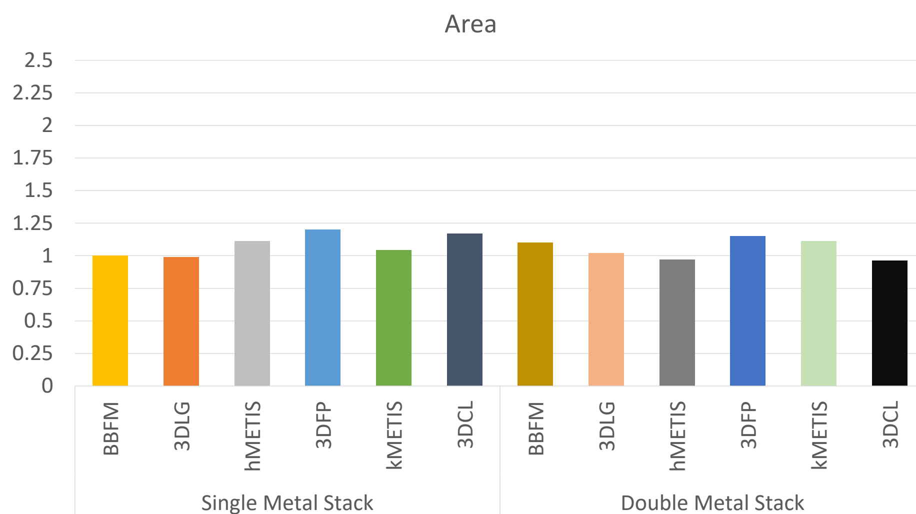


Figure 6.11 Area Comparison Results

The reason that we used the unconstrained flavor of our 3DLG, instead of the constrained 3DLG, is to investigate the impact of the number of vertical vias on the Power, Performance, and Area results. In general 3DLG presents very good results for all the examined metrices, as Figures 6.8 6.9 6.9 6.11 present. However, due to the fact that it is vertical vias agnostic, the number of vertical vias is almost two times larger than the results of the BBFM partitioning approach, as Figure 6.13 presents. It is clear that the unconstrained flavor of the 3DLG approach is compatible only to 3D integration techniques that allows high vertical interconnection density, as the monolithic 3D. Besides the Unconstrained 3DLG approach, the constrained version of the 3DLG can be used to bound the number of vertical vias, as

Institutional Repository - Library & Information Centre - University of Thessaly
30/12/2021 10:19:35 EET - 137.108.70.14

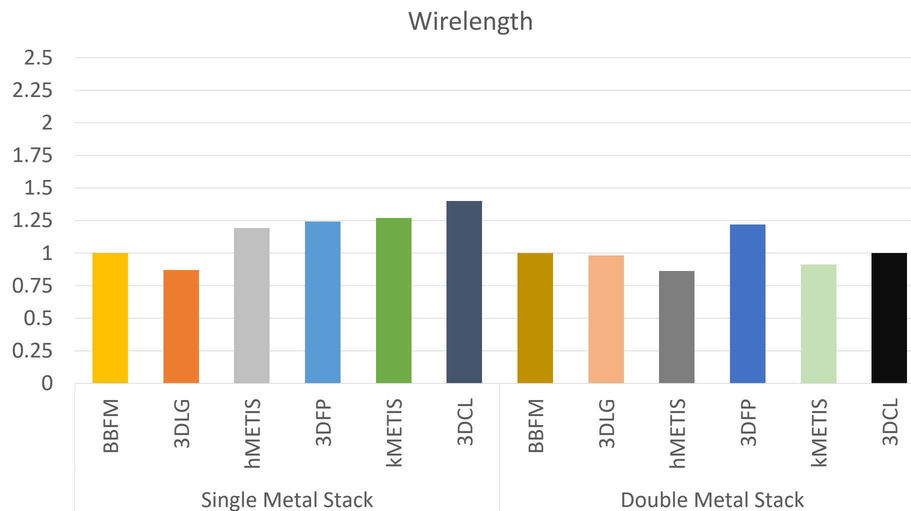


Figure 6.12 Wirelength Comparison Results

described in Section 6.2. This is the only examined approach that allows the user to set a target number of vias. This is a very important feature specifically for the via first TSV formation, where the number of vertical vias is predefined.

What is more, Figure 6.13 presents that the approach that leads to the smaller number of vertical vias, *i.e.* vertical interconnections, is the BBFM approach. This approach results in few vertical vias as its target is to minimize the number of vertical interconnections taking the connections inside each bin into account. Similarly, the hMETIS approach results in few vertical interconnections (1.1 times greater than BBFM), as both approaches, *i.e.* BBFM and hMETIS, perform partitioning to reduce the imber of the vias. The key difference between the two partitioning approaches in that BBFM starts from a placed design, and performs partitioning in each bin of the placed design, while the hMETIS performs graph partitioning after performing clustering to reduce the partitioning complexity. It is worth mentioning that the METIS tool, provides several heuristics for partitioning execution time and quality optimizations. To reduce the execution time of partitioning, METIS uses these heuristics, but they influence the final result. Moreover, it is worth pointing out that both Single and Double metal stacks present the same number of vertical vias, as all the approaches are agnostic of the vertical distance of the tiers, except for the 3DLG.

Figure 6.14 presents the runtime comparison of each approach, in terms of the execution time that is needed to assign each standard cell (for Fine-Grained approaches) or each block of standard cells (for Block-Based approaches). These runtimes do not include the P&R execution time. Similar to the vertical vias number, both Single metal stack and Double metal stack result in the same (or similar for 3DLG) tier assignment runtimes. As Figure 6.14

Institutional Repository - Library & Information Centre - University of Thessaly
30/12/2021 10:19:35 EET - 137.108.70.14

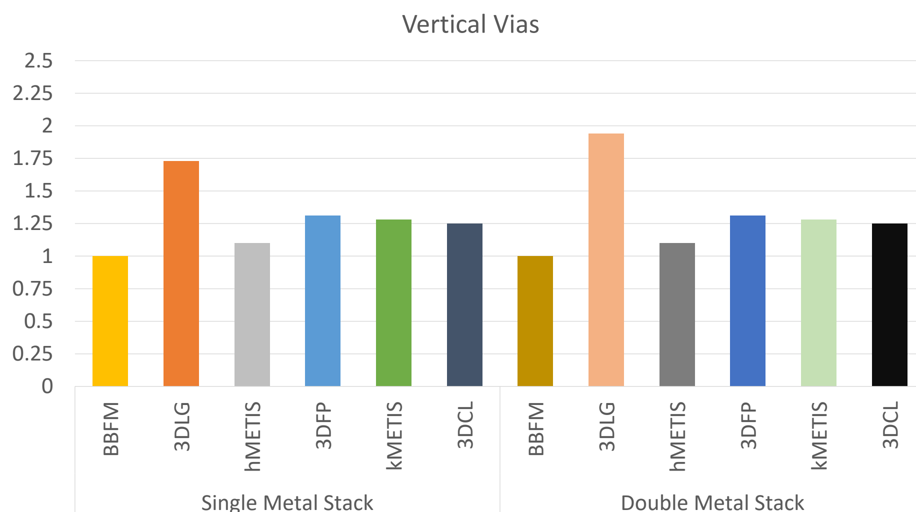


Figure 6.13 3D Vertical Vias Comparison Results

presents, 3DLG is the fastest tier assignment approach as almost half of the time is required to assign all the standard cells to tiers, compared to the baseline approach *i.e.* BBFM. In contrary to this, 3DFP is the most time consuming because this algorithm first searches for the best balanced assignment and then performs optimization moves, by using a simulated annealing technique, to reduce the inter-tier wirelength.

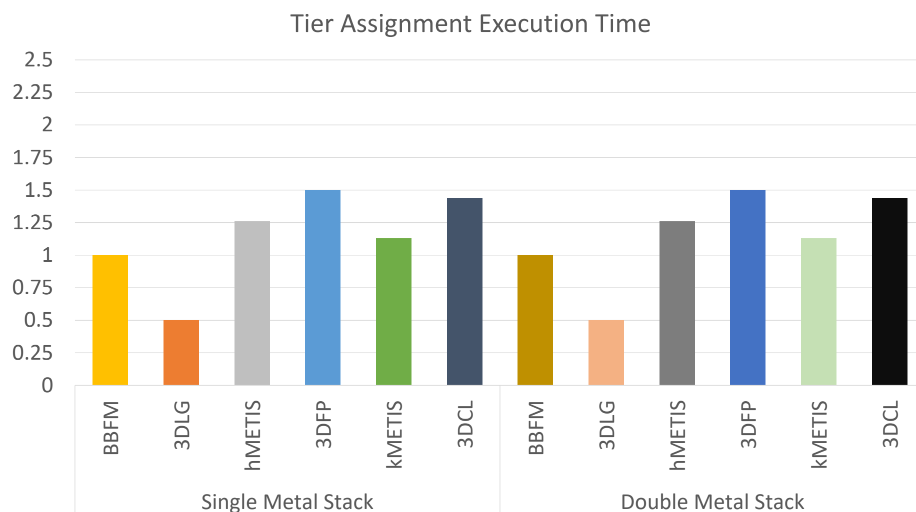


Figure 6.14 Tier Assignment Execution Time Comparison Results

Institutional Repository - Library & Information Centre - University of Thessaly
30/12/2021 10:19:35 EET - 137.108.70.14

Chapter 7

Conclusions and Future Work

In this work, we investigated different approaches and methodologies for 3D ICs integration. The examined approaches aim to solve one of the first steps of the 3D integration, the tier assignment problem. Tier assignment is a major step in the physical design flow, as it influences the quality of the final 3D IC. Therefore, the decision to assign each functional block or standard cell to a specific tier must be made very precisely.

Tier assignment can be performed in different ways, so we categorized the tier assignment strategies in two categories, *i.e.* the Block-Based and the Fine-Grained tier assignment approaches. The difference between the two categories is the level of circuit abstraction. The former either utilizes the functional blocks of the circuit or it creates blocks of standard cells automatically to take advantage of the flat connectivity of the circuit. The latter perform tier assignment at the gates' level, where the assignment complexity is greater, but they are more promising, since they optimize the assignment of each standard cell, instead of blocks.

Today, Block-Based approaches are used for the heterogeneous systems, *i.e.* systems with significant different device technologies as *e.g.* CMOS and memories. The level of abstraction of this category provides the capability to directly separate the functional blocks to different tiers, for example by using a 3D floorplanning approach. In this work, we developed a slicing floorplanning algorithm that is able to create almost perfect area balanced tiers, while it attempts to minimize the 2D wirelength. To reduce the dependency of the design architecture, *i.e.* the separation of the design into functional blocks, we investigated different approaches to create blocks of standard cells automatically, by taking the connectivity of the flat design into account. So, we developed a clustering algorithm able to create blocks of standard cells which aims to reduce the connectivity of the blocks and the area balance ratio between them. For more detailed investigation of clustering approach benefits, we also used the multi-level partitioning tool METIS, commonly used for 3D ICs tier assignment. This tool operates in two stages, firstly it performs clustering in order to reduce the circuit

Institutional Repository - Library & Information Centre - University of Thessaly
30/12/2021 10:19:35 EET - 137.108.70.14

complexity, and then performs partitioning to minimize the connectivity of the partitions. So, we used the kMETIS flavor to create area balanced partitions with minimum connectivity between the partitions, that are used as blocks in our flow.

Besides the fact that Block-Based approaches are easily applicable to the tier assignment problem, due to their high level of granularity, fine-grained approaches are more promising for better results. So, we investigated Fine-Grained tier assignment approaches, that can be used on very large designs, and compared them to the Block-Based approaches. More specifically, we implemented a Bin-Based partitioning approach that firstly divides core area into bins and then performs partitioning of the standard cells to assign each cell to a tier. This divide-and-conquer approach provides us the ability to solve the assignment problem very fast, compared to a traditional flat partitioning. Moreover, we developed the first 3D legalizer, that starts from a flat 2D placement with overlapping standard cells, while resulting in a legal 3D placement. Thus, each standard cell is assigned to the position that leads to better results. This position may be in any tier, while this approach is able to take several parameters as cost function into account, (*e.g.* wirelength, number of vertical vias, hotspots), due to its sequential nature. We investigated two flavors of the 3D legalizer, the constrained and unconstrained, where the target is to upper bound the number of vertical vias. So, constrained 3D legalizer is more suitable when the number of vias is restricted, while the unconstrained is more suitable when the number of vias does not have significant impact on the solution, as in monolithic 3D.

Additionally, we investigated the impact of the 3D metal stacks on the final Power, Performance, and Area results. So, we compared two metal stacks, *i.e.* the Single metal stack and the Double metal stack (or Face-to-Face). The former assigns the cells in the first and third metal layers for the bottom and top tier respectively, while the latter in the first and twelfth metal layers, repressively. The Single metal stack provides higher density, however it leads to higher routing congestion, while Double metal stack result in less routing congestion, but it can result in greater wirelength. The selection of the metal stack must take the tier assignment approach into consideration, as each metal stack has characteristics that are more suitable for some approaches. For example, the Single metal stack results in less wirelength and the number of vertical vias for the 3D legalizer, as this approach exploits the distance of the tiers while it does not group the cells to create strongly connected groups just like the Block-Based approaches do.

This dissertation focused on exploring the advantages and disadvantages of different assignment approaches. However, we it did not take several parameters which influence the 3D integration into consideration. As a result, our future plans on the research presented in this dissertation is to extend it towards the following areas:

Institutional Repository - Library & Information Centre - University of Thessaly
30/12/2021 10:19:35 EET - 137.108.70.14

- Extend the examined approaches to support thermal parameters. Based on the thermal profile of the design, the blocks or standard cell must be assigned to the tiers and positions that resolves hot-spots.
- One more essential parameter is circuit delay. The examined parameters does not consider the critical paths, that could cross different tiers. A future goal is to identify the critical paths and force them to be entirely in a single tier. Moreover, tier assignment should take wire and gate delay into account, for better PPA results.
- Last but not least, it is important to investigate the examined approaches in smaller technology nodes, where the wire delay is dominant. In this way, the results will be more realistic as the 45nm technology node that has been used in this dissertation did not reveal all the benefits of the 3D integration.

To create a 3D IC, it is important to note that 3D EDA tools must be developed to natively support the vertical integration. The transitional 2D tools can be used only for some steps of the flow, however it is almost infeasible to be used for massive 3D IC production.

Institutional Repository - Library & Information Centre - University of Thessaly
30/12/2021 10:19:35 EET - 137.108.70.14

Institutional Repository - Library & Information Centre - University of Thessaly
30/12/2021 10:19:35 EET - 137.108.70.14

Bibliography

[1] Gordon E Moore et al. Cramming more components onto integrated circuits, 1965.

[2] Banqiu Wu and Ajay Kumar. Extreme ultraviolet lithography and three dimensional integrated circuit—A review. *Applied Physics Reviews*, 1(1):011104, 2014.

[3] Vasilis F Pavlidis, Ioannis Savidis, and Eby G Friedman. *Three-dimensional integrated circuit design*. Newnes, 2017.

[4] Daniel Lu and CP Wong. *Materials for advanced packaging*, volume 181. Springer, 2009.

[5] Gary Charles and Paul D Franzon. Comparison of TSV-based PDN-design effects using various stacking topology methods. In *2012 IEEE 21st Conference on Electrical Performance of Electronic Packaging and Systems*, pages 83–86. IEEE, 2012.

[6] Antonis Papanikolaou, Dimitrios Soudris, and Riko Radojcic. *Three dimensional system integration: IC stacking process and design*. Springer Science & Business Media, 2010.

[7] April K Andreas and Isabel Beichl. Estimating the work in integer partitioning. *Computing in Science & Engineering*, 5(1):48–56, 2003.

[8] Nikolaos Sketopoulos, Christos Sotiriou, and Vasilis Pavlidis. Metal stack and partitioning exploration for monolithic 3D ICs. In *2020 IEEE Computer Society Annual Symposium on VLSI (ISVLSI)*, pages 398–403. IEEE, 2020.

[9] Idris Kaya, Silke Salewski, Markus Olbrich, and Erich Barke. Wirelength reduction using 3-D physical design. In *International Workshop on Power and Timing Modeling, Optimization and Simulation*, pages 453–462. Springer, 2004.

[10] Moongon Jung, Taigon Song, Yang Wan, Young-Joon Lee, Debabrata Mohapatra, Hong Wang, Greg Taylor, Devang Jariwala, Vijay Pitchumani, Patrick Morrow, et al. How to reduce power in 3D IC designs: A case study with OpenSPARC T2 core. In *Proceedings of the IEEE 2013 Custom Integrated Circuits Conference*, pages 1–4. IEEE, 2013.

[11] Nikolaos Sketopoulos, Christos Sotiriou, and Stavros Simoglou. Abax: 2d/3d legaliser supporting look-ahead legalisation and blockage strategies. In *Proceedings of the 2018 Design Automation and Test In Europe (DATE)*, pages 1469–1472. IEEE, 2018.

[12] Aida Todri-Sanial and Chuan Seng Tan. *Physical Design for 3D Integrated Circuits*. CRC Press, 2017.

Institutional Repository - Library & Information Centre - University of Thessaly
30/12/2021 10:19:35 EET - 137.108.70.14

[13] Synopsys 3D IC Tools, April, 2018.

[14] Cadence. Cadence Innovus, 2019. https://cadence.com/.

[15] Laung-Terng Wang, Yao-Wen Chang, and Kwang-Ting Tim Cheng. *Electronic design automation: synthesis, verification, and test*. Morgan Kaufmann, 2009.

[16] Vasilis F Pavlidis and Eby G Friedman. *Three-dimensional Integrated Circuit Design*. Morgan Kaufmann, 2010.

[17] Hiroshi Murata, Kunihiro Fujiyoshi, Shigetoshi Nakatake, and Yoji Kajitani. VLSI module placement based on rectangle-packing by the sequence-pair. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 15(12):1518–1524, 1996.

[18] Charles J Alpert, Dinesh P Mehta, and Sachin S Sapatnekar. *Handbook of algorithms for physical design automation*. CRC press, 2008.

[19] Wern-Jieh Sun and Carl Sechen. Efficient and effective placement for very large circuits. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*.

[20] Jarrod A Roy, David A Papa, Saurabh N Adya, Hayward H Chan, Aaron N Ng, James F Lu, and Igor L Markov. Capo: robust and scalable open-source min-cut floorplacer. In *Proceedings of the 2005 international symposium on Physical design*, pages 224–226. ACM, 2005.

[21] Taraneh Taghavi, Xiaojian Yang, et al. Dragon2005: Large-scale mixed-size placement tool. In *Proceedings of the 2005 international symposium on Physical design*, pages 245–247. ACM, 2005.

[22] Ameya R Agnihotri, Satoshi Ono, and Patrick H Madden. Recursive bisection placement: feng shui 5.0 implementation details. In *Proceedings of the 2005 international symposium on Physical design*, pages 230–232. ACM, 2005.

[23] Tung-Chieh Chen, Zhe-Wei Jiang, Tien-Chang Hsu, Hsin-Chen Chen, and Yao-Wen Chang. NTUplace3: An analytical placer for large-scale mixed-size designs with preplaced blocks and density constraints. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 27(7):1228–1240, 2008.

[24] Jingwei Lu, Pengwen Chen, Chin-Chih Chang, Lu Sha, Dennis J Huang, Chin-Chi Teng, Chung-Kuan Cheng, et al. ePlace: Electrostatics based placement using Nesterov's method. In *Proceedings of the 51st Annual Design Automation Conference*, pages 1–6. ACM, 2014.

[25] David S Kershaw. The incomplete cholesky—conjugate gradient method for the iterative solution of systems of linear equations. *Journal of Computational Physics*, 26(1):43–65, 1978.

[26] Andrew B Kahng, Sherief Reda, and Qinke Wang. Aplace: A general analytic placement framework. In *Proceedings of the 2005 international symposium on Physical design*, pages 233–235. ACM, 2005.

Institutional Repository - Library & Information Centre - University of Thessaly
30/12/2021 10:19:35 EET - 137.108.70.14

[27] William C Naylor, Ross Donelly, and Lu Sha. Non-linear optimization system and method for wire length and delay optimization for an automatic electric circuit placer, October 9 2001. US Patent 6,301,693.

[28] Gi-Joon Nam and Jingsheng Jason Cong. *Modern circuit placement: best practices and results*. Springer Science & Business Media, 2007.

[29] Jürgen M Kleinhans, Georg Sigl, Frank M Johannes, and Kurt J Antreich. GORDIAN: VLSI placement by quadratic programming and slicing optimization. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 10(3):356–365, 1991.

[30] Ulrich Brenner, Markus Struzyna, and Jens Vygen. BonnPlace: Placement of leading-edge chips by advanced combinatorial algorithms. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 27(9):1607–1620, 2008.

[31] Peter Spindler, Ulf Schlichtmann, and Frank M Johannes. Kraftwerk2—A fast force-directed quadratic placement approach using an accurate net model. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 27(8):1398–1411, 2008.

[32] Andrew Kennings and Kristofer P Vorwerk. Force-directed methods for generic placement. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 25(10):2076–2087, 2006.

[33] Bo Hu and Malgorzata Marek-Sadowska. Multilevel fixed-point-addition-based vlsi placement. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 24(8):1188–1203, 2005.

[34] Fan Mo, Abdallah Tabbara, and Robert K Brayton. A force-directed macro-cell placer. In *Proceedings of the 2000 IEEE/ACM international conference on Computer-aided design*, pages 177–181. IEEE Press, 2000.

[35] Natarajan Viswanathan, Min Pan, and Chris Chu. FastPlace 3.0: A fast multilevel quadratic placement algorithm with placement congestion control. In *Proceedings of the 2007 Asia and South Pacific Design Automation Conference*, pages 135–140. IEEE Computer Society, 2007.

[36] George Karypis and Vipin Kumar. Multilevel k-way hypergraph partitioning. *VLSI design*, 11(3):285–300, 2000.

[37] Gi-Joon Nam, Sherief Reda, Charles J Alpert, Paul G Villarrubia, and Andrew B Kahng. A fast hierarchical quadratic placement algorithm. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 25(4):678–691, 2006.

[38] Johann Knechtel and Jens Lienig. Physical Design Automation for 3D Chip Stacks: Challenges and Solutions. In *Proceedings of the 2016 on International Symposium on Physical Design*, pages 3–10. ACM, 2016.

[39] Karthik Balakrishnan, Vidit Nanda, Siddharth Easwar, and Sung Kyu Lim. Wire congestion and thermal aware 3D global placement. In *Proceedings of the 2005 Asia and South Pacific Design Automation Conference*, pages 1131–1134. ACM, 2005.

Institutional Repository - Library & Information Centre - University of Thessaly
30/12/2021 10:19:35 EET - 137.108.70.14

[40] Charles Chiang and Subarna Sinha. The road to 3D EDA tool readiness. In *Proceedings of the 2009 Asia and South Pacific Design Automation Conference*, pages 429–436. IEEE Press, 2009.

[41] Brent Goplen and Sachin Sapatnekar. Efficient thermal placement of standard cells in 3D ICs using a force directed approach. In *Proceedings of the 2003 IEEE/ACM international conference on Computer-aided design*, page 86. IEEE Computer Society, 2003.

[42] Renato Hentschke, Guilherme Flach, Felipe Pinto, and Ricardo Reis. 3D-vias aware quadratic placement for 3D VLSI circuits. In *VLSI, 2007. ISVLSI'07. IEEE Computer Society Annual Symposium on*, pages 67–72. IEEE, 2007.

[43] Haixia Yan, Qiang Zhou, and Xianlong Hong. Thermal aware placement in 3D ICs using quadratic uniformity modeling approach. *INTEGRATION, the VLSI journal*, 42(2):175–180, 2009.

[44] Jason Cong and Guojie Luo. A multilevel analytical placement for 3D ICs. In *Design Automation Conference, 2009. ASP-DAC 2009. Asia and South Pacific*, pages 361–366. IEEE, 2009.

[45] DP Bertsekas. Approximation procedures based on the method of multipliers. *Journal of Optimization Theory and Applications*, 23(4):487–510, 1977.

[46] Jason Cong, Jie Wei, and Yan Zhang. A thermal-driven floorplanning algorithm for 3D ICs. In *IEEE/ACM International Conference on Computer Aided Design, 2004. ICCAD-2004.*, pages 306–313. IEEE, 2004.

[47] W-L Hung, Greg M Link, Yuan Xie, Narayanan Vijaykrishnan, and Mary Jane Irwin. Interconnect and thermal-aware floorplanning for 3D microprocessors. In *7th International Symposium on Quality Electronic Design (ISQED'06)*, pages 6–pp. IEEE, 2006.

[48] Michael Healy, Mario Vittes, Mongkol Ekpanyapong, Chinnakrishnan S Ballapuram, Sung Kyu Lim, Hsien-Hsin S Lee, and Gabriel H Loh. Multiobjective microarchitectural floorplanning for 2-D and 3-D ICs. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 26(1):38–52, 2006.

[49] Young-Joon Lee and Sung Kyu Lim. On GPU bus power reduction with 3D IC technologies. In *2014 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, pages 1–6. IEEE, 2014.

[50] Pingqiang Zhou, Yuchun Ma, Zhouyuan Li, Robert P Dick, Li Shang, Hai Zhou, Xianlong Hong, and Qiang Zhou. 3D-STAF: scalable temperature and leakage aware floorplanning for three-dimensional integrated circuits. In *2007 IEEE/ACM International Conference on Computer-Aided Design*, pages 590–597. IEEE, 2007.

[51] Luciano Lavagno, Igor L Markov, Grant Martin, and Louis K Scheffer. *Electronic design automation for IC implementation, circuit design, and process technology: circuit design, and process technology*. CRC Press, 2016.

Institutional Repository - Library & Information Centre - University of Thessaly
30/12/2021 10:19:35 EET - 137.108.70.14

[52] Cadence 3D IC Tools, April, 2018.

[53] Shreepad A Panth, Kambiz Samadi, Yang Du, and Sung Kyu Lim. Design and cad methodologies for low power gate-level monolithic 3d ics. In *Proceedings of the 2014 International Symposium on Low-Power Electronics and Design (ISLPED)*, pages 171–176. ACM, 2014.

[54] Bon Woong Ku, Peter Debacker, Dragomir Milojevic, Praveen Raghavan, Diederik Verkest, Aaron Thean, and Sung Kyu Lim. Physical design solutions to tackle feol/beol degradation in gate-level monolithic 3d ics. In *Proceedings of the 2016 International Symposium on Low Power Electronics and Design*, pages 76–81. ACM, 2016.

[55] Kyungwook Chang, Abhishek Koneru, Krishnendu Chakrabarty, and Sung Kyu Lim. Design automation and testing of monolithic 3D ICs: Opportunities, challenges, and solutions. In *Computer-Aided Design (ICCAD), 2017 IEEE/ACM International Conference on*, pages 805–810. IEEE, 2017.

[56] Sai Surya Kiran Pentapati, Da Eun Shim, and Sung Kyu Lim. Logic Monolithic 3D ICs: PPA Benefits and EDA Tools Necessary. In *Proceedings of the 2019 on Great Lakes Symposium on VLSI*, pages 445–450, 2019.

[57] Kyungwook Chang, Saurabh Sinha, Brian Cline, Raney Southerland, Michael Doherty, Greg Yeric, and Sung Kyu Lim. Cascade2D: A design-aware partitioning approach to monolithic 3D IC with 2D commercial tools. In *Computer-Aided Design (ICCAD), 2016 IEEE/ACM International Conference on*, pages 1–8. IEEE, 2016.

[58] Bon Woong Ku, Kyungwook Chang, and Sung Kyu Lim. Compact-2d: A physical design methodology to build commercial-quality face-to-face-bonded 3d ics. In *Proceedings of the 2018 International Symposium on Physical Design*, pages 90–97. ACM, 2018.

[59] Charles M Fiduccia and Robert M Mattheyses. A linear-time heuristic for improving network partitions. In *Papers on Twenty-five years of electronic design automation*, pages 241–247. ACM, 1988.

[60] Richard E Korf. A complete anytime algorithm for number partitioning. *Artificial Intelligence*, 106(2):181–203, 1998.

[61] DF Wong and CL Liu. A new algorithm for floorplan design. In *23rd ACM/IEEE Design Automation Conference*, pages 101–107. IEEE, 1986.

[62] JS Arora. 15—Discrete Variable Optimum Design Concepts and Methods. *Introduction to Optimum Design, 2nd ed.; Arora, JS, Ed*, pages 513–530.

[63] METIS - Serial Graph Partitioning and Fill-reducing Matrix Ordering. http://glaros.dtc.umn.edu/gkhome/views/metis. Accessed: 2021-07-01.

[64] George Karypis, Rajat Aggarwal, Vipin Kumar, and Shashi Shekhar. Multilevel hypergraph partitioning: Applications in VLSI domain. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 7(1):69–79, 1999.

Institutional Repository - Library & Information Centre - University of Thessaly
30/12/2021 10:19:35 EET - 137.108.70.14

[65] Charles M Fiduccia and Robert M Mattheyses. A linear-time heuristic for improving network partitions. In *19th Design Automation Conference*, pages 175–181. IEEE, 1982.

[66] Shreepad Panth, Kambiz Samadi, Yang Du, and Sung Kyu Lim. Placement-driven partitioning for congestion mitigation in monolithic 3d ic designs. In *Proceedings of the 2014 International Symposium on Physical Design (ISPD)*, 2014.

[67] Siroos Madani and Magdy Bayoumi. In *2017 18th International Workshop on Microprocessor and SOC Test and Verification (MTV)*, pages 57–61. IEEE, 2017.

[68] Duckhwan Kim and Saibal Mukhopadhyay. Partitioning methods for interface circuit of heterogeneous 3-D-ICs under process variation. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 24(5):1626–1635, 2015.

[69] Sabyasachee Banerjee, Subhashis Majumder, and Bhargab B Bhattacharya. A graph-based 3D IC partitioning technique. In *2014 IEEE computer society annual symposium on VLSI*, pages 613–618. IEEE, 2014.

[70] Sadiq M Sait, Feras Chikh Oughali, and Mohammed Al-Asli. Design partitioning and layer assignment for 3D integrated circuits using tabu search and simulated annealing. *Journal of applied research and technology*, 14(1):67–76, 2016.

[71] Nikolaos Sketopoulos, Christos P Sotiriou, and Vasileios Samaras. Investigation and Trade-offs in 3DIC Partitioning Methodologies. In *Proceedings of the 2019 on Great Lakes Symposium on VLSI*, pages 451–455, 2019.

[72] Shreepad Panth, Kambiz Samadi, Yang Du, and Sung Kyu Lim. Placement-driven partitioning for congestion mitigation in monolithic 3D IC designs. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 34(4):540–553, 2015.

[73] New York nanoCAS Lab, Stony Brook University. nanocas two-tier 3d 45nm nangate converted library, 2018.

[74] OpenCores. OpenCores Open IP Community, 2019. https://opencores.org/.

[75] Chen Yan and Emre Salman. Mono3D: Open source cell library for monolithic 3-D integrated circuits. *IEEE Transactions on Circuits and Systems I: Regular Papers*, 65(3):1075–1085, 2017.

Institutional Repository - Library & Information Centre - University of Thessaly
30/12/2021 10:19:35 EET - 137.108.70.14