12-2021

Design, Extraction, and Optimization Tool Flows and Methodologies for Homogeneous and Heterogeneous Multi-Chip 2.5D Systems

MD Arafat Kabir
University of Arkansas, Fayetteville

Design, Extraction, and Optimization Tool Flows and Methodologies
for Homogeneous and Heterogeneous Multi-Chip 2.5D Systems

A thesis submitted in partial fulfillment
of the requirements for the degree of
Master of Science in Computer Engineering

by

MD Arafat Kabir
Bangladesh University of Engineering and Technology
Bachelor of Science in Electrical and Electronic Engineering, 2017

December 2021
University of Arkansas

This thesis is approved for recommendation to the Graduate Council.

Yarui Peng, Ph.D.
Thesis Director

Alexander H. Nelson, Ph.D.
Committee Member

David Andrews, Ph.D.
Committee Member

Abstract

Chip and packaging industries are making significant progress in 2.5D design as a result of increasing popularity of their application. In advanced high-density 2.5D packages, package redistribution layers become similar to chip Back-End-of-Line routing layers, and the gap between them scales down with pin density improvement. Chiplet-package interactions become significant and severely affect system performance and reliability. Moreover, 2.5D integration offers opportunities to apply novel design techniques. The traditional die-by-die design approach neither carefully considers these interactions nor fully exploits the cross-boundary design opportunities.

This thesis presents chiplet-package cross-boundary design, extraction, analysis, and optimization tool flows and methodologies for high-density 2.5D packaging technologies. A holistic flow is presented that can capture all parasitics from chiplets and the package and improve system performance through iterative optimizations. Several design techniques are demonstrated for agile development and quick turn-around time. To validate the flow in silicon, a chip was taped out and studied in TSMC 65nm technology. As the holistic flow cannot handle heterogeneous technologies, in-context flows are presented. Three different flavors of the in-context flow are presented, which offer trade-offs between scalability and accuracy in heterogeneous 2.5D system designs. Inductance is an inseparable part of a package design. A holistic flow is presented that takes package inductance into account in timing analysis and optimization steps. Custom CAD tools are developed to make these flows compatible with the industry standard tools and the foundry model. To prove the effectiveness of the flows several design cases of an ARM Cortex-M0 are implemented for comparitive study.

Acknowledgements

First, I would like to express my great appreciation to my research and thesis advisor, Dr. Yarui Peng, for his mentorship, direction, and encouragement. He has spent more than three years teaching me through discussion, offering me challenges, and pushing me beyond my limits. I want to thank him for the knowledge I have gained from his teaching, making me actively participate in his research and academic work, the advice on the development of my research, and the completion of this thesis.

I would like to thank Dr. David Andrews and Dr. Alexander Nelson, for taking time out of their schedules to participate in the committee for this thesis. I would like to thank Dr. Nelson for advising me during the preparation of this thesis; teaching me when I took his course; for his communication, cooperation, and understanding the problems of a graduate student when I was serving as a teaching assistant for his course. I would like to thank Dr. Andrews for teaching and helping me in the completion of the required courses for this degree. I am grateful for all the help and cooperation I have received from them when I need them most.

I want to express my gratitude to the National Science Foundation (NSF) for supporting the project that led to this thesis. NSF provided with the funding and work that allowed me to pursue my graduate degree. I also want to thank Dr. Dusan Petranovic for all his support and advice in my research. His industry affiliation, experience, and insights have helped come up with better approaches and solutions in my research. I would like to thank my colleague, Imam Al Razi, for supporting me and working with me throughout the time of this research. Imam is one of the hardest working and most persistent students I have ever worked with. He has been a mentor, an elder brother, and mostly a caring friend, who has helped make my journey easier.

Finally, I would like to thank my family and friends who have all supported me in every step of the way to achieving this goal. It would not have been possible for me to finish this degree without all of them. With far too many names to list, I would like to tell them how thankful I am for their care, support, and motivation during this journey.

Dedication

To my mother, who always supported me to pursue my dreams

To my father, who always believed in me

To my brother, who showed me how not to lose hope in difficult times

To my wife, who has been by my side supporting me whenever I stumbled

This thesis and all of my achievements are the results of the care and support I have received from you all. Without your continued belief in me to succeed, I would not be where I am today.

I LOVE YOU ALL

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

The demands for increased functionality and performance in emerging technologies, such as 5G, Artificial Intelligence, and high performance computing have pushed modern chips to the brink of the reticle limit. The industry responses with a modular design approach, in which a large system is broken down into smaller modules and then integrated as a whole system on the package. Traditionally, a Printed-Circuit-Board (PCB) is used as the integration platform. PCBs are cheap and easy to design. However, they suffer from high inductance and capacitance because of long and wide traces, thereby limiting bandwidth and increasing power loss. In the past few decades, the electronic packaging technology has been through substantial development. Starting from the dual in-line package (DIP), the packaging technology has evolved through quad flat package (QFP), ball-grid array (BGA), chip-scale package (CSP), etc. and today offers high-density, high-bandwidth, and energy-efficient 2.5D and 3D packaging solutions

1.1 Introduction to 2.5D Systems

A multi-chip 2.5D system consists of multiple dies (chiplets), which are interconnected through redistribution layers (RDLs) on an interposer. Fig. 1 (b) and (c) show examples of two-chiplet 2.5D systems. Fig. 1 (b) shows a flip-chip package, while Fig. 1 (c) shows a bump-less wafer-level package (WLP). A 2.5D integration package may use different materials like glass, slilicon, and polymer as interposer [1]. 2.5D systems have overcome several drawbacks of traditional PCB-based systems and are providing energy efficient inter-die transmission and high-density integration options. With transistor scaling saturated, these 2.5D designs are becoming popular in high-density applications like mobile phones and tablets [2]. Moreover, 2.5D packaging enables heterogeneous integration [3, 4] and high-bandwidth inter-die communication [5]. It also offers promising hardware security applications [6, 7]. The increasing interests are driving the industry to develop compact and high-performance 2.5D packaging solutions [8]. In the last few years, the

industry has developed several 2.5D integration technologies like eWLB [9], SWIFT [10], and InFO [11]. In every iteration of these technologies, the packages wires become thinner and denser, bringing chips and packages very close with a reduced pad size.

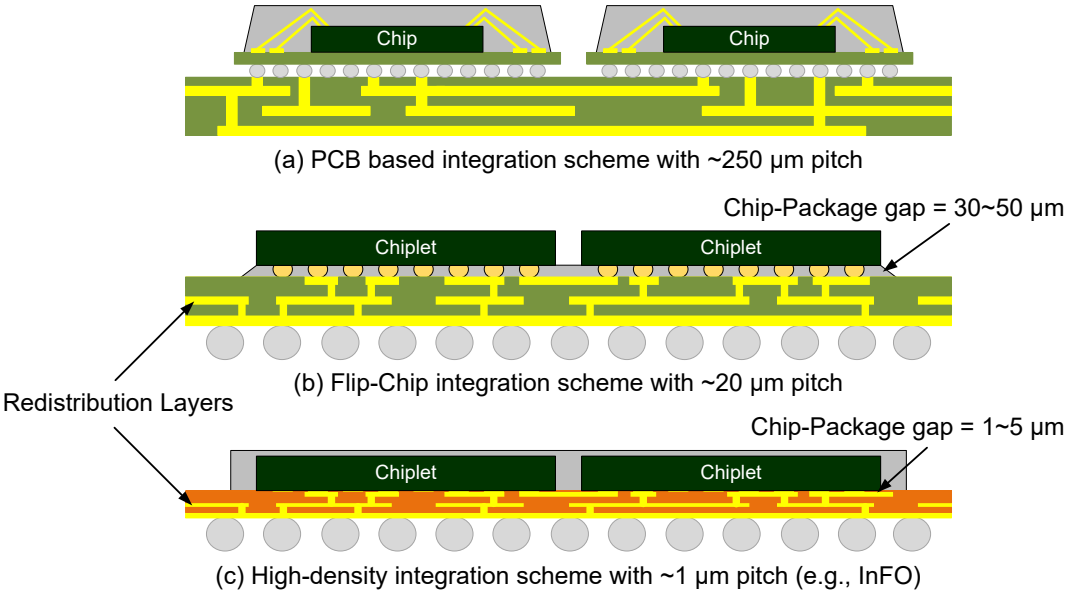


Figure 1: 2.5D integration schemes: (a) PCB based system, (b) flip-chip with an organic interposer, (c) high-density integration scheme such as wafer-level-packaging

1.2 Need for a Cross-Boundary Flow

In the traditional flow, 2.5D systems are designed in a die-by-die (DbD) approach where each chiplet is designed independently as a single unit, and then all chiplets are mounted on the package as a complete system. The analysis and optimization of chiplets and the package are also conducted separately without consideration of the interactions between them [12, 13]. Fig. 2 illustrates this traditional flow in which chiplets and the package never interact with each other until they are fabricated and assembled. During package design, a chiplet is approximated as a ground mesh or plane. In this approach, it is possible to achieve the shortest possible turn-around time using off-the-shelf chiplets in a plug-and-play manner [12]. This flow is sufficient when the gap between chiplets and the package is large enough to make the interactions between them minimal. As shown in Fig. 1(b), in flip-chip WLP, this gap is around 30µm~50µm. In such

integration technologies, the traditional flow can be used without any critical problem. However, due to the industry's aggressive development and the adoption of bumpless contact pads [4], this gap is decreasing rapidly [14, 15, 16, 17]. Within a few years, this gap is reduced from tens of µm to 1.5 µm [16]. At such a small separation, significant capacitive and inductive coupling is expected between chip and package routing layers. To handle such high-density integration schemes, a cross-boundary design flow is required, which can capture these interactions during design and optimization steps of both chiplet and package.

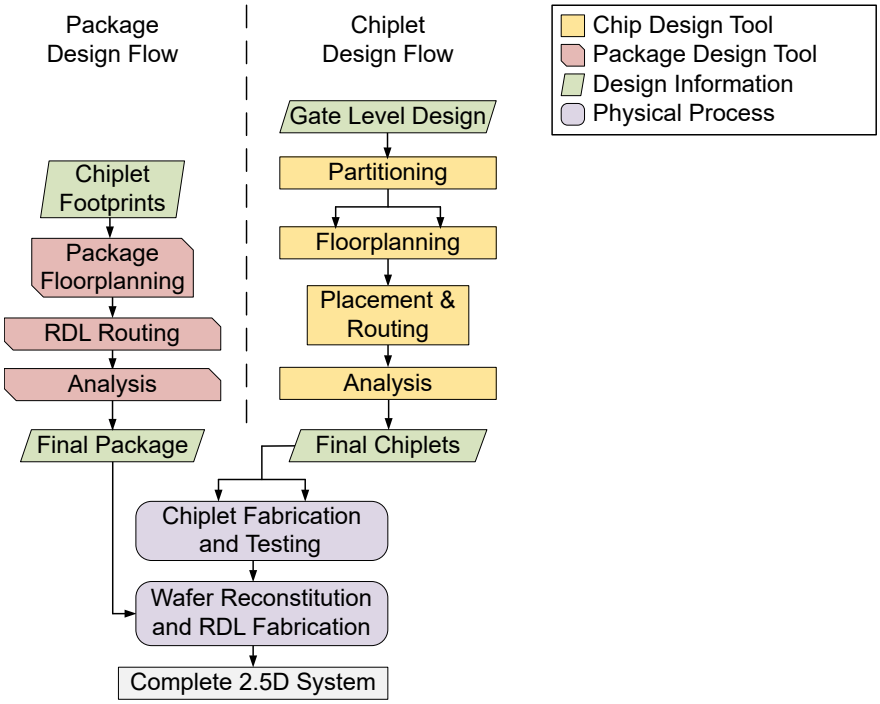


Figure 2: The traditional Die-by-Die design flow of a 2.5D system

In the die-by-die approach, the complete system is not considered as a whole. Therefore, it is not possible to obtain a globally optimized system, though individual chiplets may perform well. Because of the pin-dominated nature of package routing, it can get overly complicated, introducing unnecessary package overhead due to detours if chiplet pins are not planned properly. As all the chiplets work together as a single system, timing optimizations need to be performed at the system level. While planning the package, it may be necessary to rearrange the chiplets pin configurations to obtain a compact package routing to minimize package wire delays. The

post-design analysis tools need to consider chiplets and package interactions to avoid signal reliability issues, potentially causing system failure.

In addition, some designs are intrinsically very small in area and power budget, such as IoT devices. Having a large/high-performance IO system will create too much overhead for such system to be implemented with multiple chiplets. One solution is to allow highly-customized IO interface to be used between chiplets which can be simplified into a few standard cells. However, as these cells are not designed for driving long RDL wires with many technology variations, parasitics and STA analysis must be performed very carefully to avoid potential violations. To overcome these challenges, it is essential to design a CAD method for low-cost IO systems that reduces timing and power overheads but still captures all couplings between the chiplet and package to ensure all design constraints are met.

1.3 Existing Work

In recent years, 2.5D system design methodologies have gained significant attention of both industry and academic researchers. There are existing studies that investigate different aspects of 2.5D system design. The design methodologies are investigated from different perspectives like IP-reuse, active interposer-based integration, RDL routing, Power Delivery Network (PDN), thermal issues, security, etc. This section outlines some of the recently published work in this area, which are most relevant with this thesis.

1.3.1 Flows for IP-reuse and Active Interposer

2.5D integration platforms enable plug-and-play design approach through chiplet-based IP reuse. A complete system can be built through integration of off-the-shelf homogeneous or heterogeneous chiplets on the interposer in a very short time. A vertically integrated design flow for chiplet creation and integration utilizing 2.5D platforms was presented in [12]. They proposed a new protocol called Hybrid-Link for communication between chiplets that can be utilized in the plug-and-play design approach.

The 2.5D integration platform can utilize either active or passive interposer. A passive interposer contains only inter-chiplet interconnects and possibly some passive elements like capacitor, inductor, etc. An active interposer can have logic devices in addition to interconnects and passive devices. A design flow for active interposer-based 2.5D systems was presented in [18]. The flow utilizes chip design and analysis tools to design the active elements in the interposer.

Both of the works presented complete flows for 2.5D system design, which accomplishes their respective design goals, IP-reuse and active interposer design. They also explored some co-design approaches. However, none of the flows tries to capture the chiplet-package cross-boundary interactions and study their impact on the system performance.

1.3.2 RDL Routing Methodologies

Though 2.5D integration is a packaging technology, it uses ultra-fine pitch similar to the backend-of-the-line (BEOL) processes. As a result, traditional packaging tools cannot handle the complexity. Though chip design tools can handle the design complexity, existing methodologies for chip routing are not suitable for certain package design features, like any-angle routing, variable pad and trace width, irregular pad structures, etc. As a result, active research is going on to address the RDL routing problem.

A unified routing framework that can handle both grid-based and gridless routing on RLDLs was proposed in [19]. This work employs the modulus-based matrix splitting iteration method (MMSIM) and a bipartite matching algorithm in their methodology. Methodologies to handle routing problems with pre-assignment and via-based multi-chip multi-layer high-density 2.5D packages with irregular pad structures was proposed in [20]. This work proposes an algorithm to perform single-layer concurrent routing and an octagonal tile model. The tile model can handle triangles, rectangles, and trapezoids with 45°, 90°, and 135°interior angles.

Both of the works proposes novel routing methodologies to tackle the RDL routing problem. These methodologies have great performance in terms of routability, total wirelength, and

runtime. However, these work lack the study of the impact of those methodologies at the system level.

1.3.3 Power Delivery and Thermal Aware Methodologies

In a 2.5D system, the power delivery network design is very important because it affects the IR-drop of chiplet designs. At high frequency, the power delivery network needs to be characterized accurately. A methodology for co-design, co-anlysis, and the system-level optimization of chiplet and interposer PDN is proposed in [21]. High-density package with closely packed chiplets suffer from thermal-based failures if the chiplets have high power densities. A thermally-aware chiplet placement methodology for heterogeneous 2.5D systems was proposed in [22]. This work proposes a simulated annealing-based placer that strategically inserts spacing between chiplets to jointly minimize the peak temperature of the overall system and the total inter-chiplet network wirelength. Both of these flows target very specific aspects of the 2.5D system design.

1.3.4 Design Flows for Security

2.5D integration platforms offer some great hardware IP security features. Some research has been conducted on this ground. A security-aware physical design flow for 2.5D systems was proposed in [6] that can prevent IP piracy. The methodology obfuscates the details of the design in the out-sourced part of the system through careful partitioning and placement. Methodologies to implement security features for network-on-chip (NoC) architectures were proposed in [18]. The security scheme implements a trustworthy communication backbone to prevent threats like snooping of communication, spoofing identifiers, malicious access or modification of data in shared memory, etc. and offers runtime monitoring of system-level memory requests.

1.4 Contributions of This Work

In this work, chiplet-package co-design and optimization flows and methodologies are presented that employ a cross-boundary strategy to design chiplets and the package together. In these flows, chiplets and the package (at least part of it) are assembled in a common design environment during planning and analysis steps for holistic consideration. This shared layout database allows exchanging necessary cross-boundary design information to capture coupling and mutual interactions, which is essential to achieve high analysis accuracy, co-optimization of the chiplets, and reliable system design.

The effectiveness and flexibility the proposed flows are illustrated through the study of several 2.5D design cases of an ARM Cortex-M0 based microcontroller system. To verify that the flows are compatible with the existing foundry model and industry standard tools, a chip was designed using one of the proposed flows in TSMC 65nm technology. The contributions listed below are the direct result of this work.

1. An ASIC-CAD-compatible holistic flow that can design, optimize, and analyze homogeneous 2.5D systems with high-density FOWLP technologies,
2. Three flavors of cross-boundary in-context flow for design, optimization, and analysis of heterogeneous 2.5D systems,
    (a) A scalable per-chiplet in-context flow,
    (b) A highly accurate per-technology in-context flow,
    (c) A timing-accurate scalable in-context flow,
3. Package inductance-aware system-level cross-boundary timing optimization methodology

1.4.1 Holistic Flow for Homogeneous Systems

In all of the existing tool flows discussed in Section 1.3, the die-by-die design, analysis, and optimization flows are employed. Some co-design is achieved through manual transfer of design

information. To achieve a globally optimized system through co-design, a tool flow with inherent exchange of chiplet-package cross-boundary design information is essential. The holistic 2.5D chiplet-package co-design and optimization flow presented here employs a cross-boundary strategy to design chiplets and the package together. In this flow, chiplets and the package are assembled in a common design environment during planning and analysis steps for holistic consideration. This shared layout database allows exchanging necessary cross-boundary design information to capture coupling and mutual interactions, which is essential to achieve high analysis accuracy, co-optimization of the chiplets, and reliable system design. This flow has been silicon verified through tape-out of a system in TSMC 65nm technology.

1.4.2 In-Context Flows for Homogeneous Systems

Since the holistic flow requires to assemble the chiplets into a unified design environment, it cannot be applied to heterogeneous systems where the device stack are different. At the present, no standard computer aided design (CAD) flows support including two different technology files into a single physical design tools. Therefore, the in-context design flows are presented, which allows an arbitrary number of chiplets in different technologies integrated with chiplet-package coupling considered altogether. It is completely compatible with all standard ASIC tools for design, extraction, and analysis. The three flavors mentioned above offers trade-off between analysis accuracy and scalability. These flavors can be utilized together to design and analyze a single system at different stage of the design process.

1.4.3 Inductance-Aware System-Level Timing Optimization

Traditionally, chip-scale interconnects are modeled using resistive and capacitive (RC) elements. Several previous studies have discussed the impact of inductive (L) elements of interconnects at the high-frequency range. It is essential to take into account the inductive behaviors of the RDL wires to ensure system reliability and signal integrity of a high-performance 2.5D system with long interconnects. A system-level analysis and optimization flow has been presented for

chiplet-package co-optimization, which takes into account the inductive effects of RDL wires on the system performance. This flow can automatically co-optimize the IO drivers and receivers between chiplets taking into account the timing overhead introduced by the inductive behavior of the RDL wires. This analysis and optimization flow can be used a a part of the holistic and in-context flows to perform inductance-aware design of both homogeneous and heterogeneous 2.5D systems.

Chapter 2

Holistic Methodologies

Apart from heterogeneous integration, 2.5D integration technology enables the chiplet design approach. A large ASIC chip can be partitioned into smaller chiplets in order to increase yield through the use of the Known-Good-Dies [23]. In such systems, to ensure reliable inter-chiplet communication, an additional stage in the pipeline, like SerDes [3, 15], would be necessary to hide IO overhead. This would require changes at the architecture level. Changes in architecture require sufficient engineering efforts and are not so quick and flexible. The traditional method is to carefully design an IO interface and an architectural change in the pipeline depth and timing requirement. This is a complicated process that takes a few months ahead for planning and implementation. Though this is not an issue for large design houses, small ASIC design companies may not have enough resources and time for such architecture exploration. In that case, a large engineering design margin needs to be left such that the IOs from different chiplets can communicate with each other within the design tolerance. Novel IO designs [24, 25] have been proposed for 2.5D systems, which will significantly reduce the IO overhead and power consumption. However, as these cells are not designed for driving long redistribution layer (RDL) wires with many technology variations, parasitic extraction and static timing analysis (STA) must be performed very carefully to avoid potential violations to the overall system performance and signal integrity issues.

To enable agile customization without the need for a complete re-design, all parts in a 2.5D system must be considered holistically as much as possible. In this chapter, a holistic chiplet-package co-design and optimization flow is presented that facilitates the exchange of chiplet-package cross-boundary design information to obtain a globally optimized system with the highest system reliability. In this co-optimization methodology, all chiplets are automatically adjusted, making trade-offs among themselves for the package overhead. The holistic flow takes care of the co-optimization parts and offers flexibility and speed to explore different chipletization

and package design schemes without touching the system architecture. Through the use of all standard libraries to design custom pin drivers, it achieves zero overhead on pipeline depth and minimizes the timing and power overhead.

Through the study of several 2.5D design cases of an ARM Cortex-M0 based microcontroller system, the effectiveness and flexibility of the flow has been illustrated. To verify the flow in silicon, a chip that is designed using the flow is taped-out and studied in TSMC 65nm technology. This chapter presents an ASIC-CAD-compatible holistic flow that can design, optimize, and analyze 2.5D systems with high-density FOWLP technologies, a study of the necessity and effectiveness of holistic extraction and STA on 2.5D systems designed in commercial technologies, illustration of design flexibility and speed offered by the holistic flow with both drop-in and pay-as-you-use design strategies, and silicon validation of the flow with a 2D/2.5D tape-out design in TSMC 65nm technology.

2.1 Design Settings and CAD Flow

In this section, the overall holistic flow is briefly introduced. As an illustration, an ARM Cortex-M0 based microcontroller system is designed with a modified TSMC 65nm PDK. The architecture and the modified settings are also presented.

2.1.1 Architecture and Chiplet Partitions

Fig. 3 shows the system architecture of the microcontroller system. It has an ARM Cortex-M0 processor core connected to the rest of the system through AMBA High-performance Bus (AHB). The AHB bus is connected to the system controller, two GPIO modules, a ROM table, the memory interface, and an Advanced Peripheral Bus (APB) sub-system. The APB sub-system consists of a watchdog timer, two simple timers, a dual-timer, UART modules, etc. The system has a total of 16KB memory divided into four 4KB banks. The memory interface is designed in a way that each bank occupies a contiguous address range. As a result, the system can operate even if some of the memory banks addressed by the upper address range are not present. RAM and

ROM macros are compiled using ARM memory compilers.

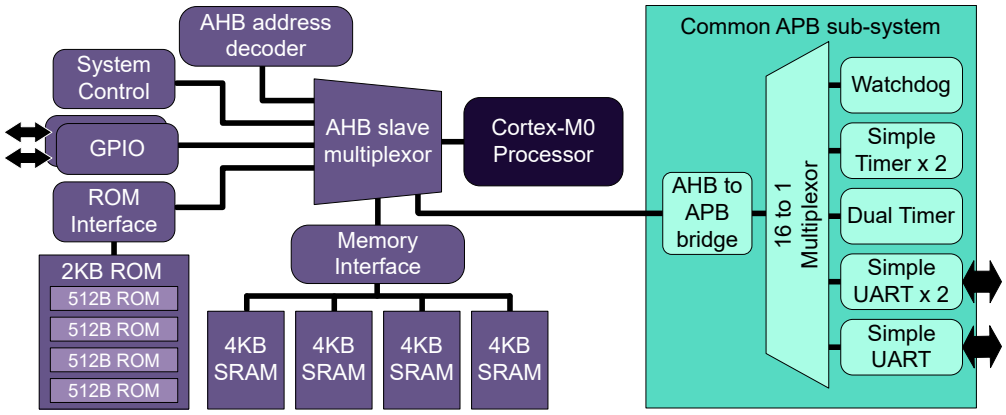


Figure 3: System architecture of the ARM Cortex-M0-based microcontroller

To implement the microcontroller as a 2.5D system, it is partitioned into two chiplets. Several partitioning algorithms and design schemes were studied to understand the impact of package wires during the partition stage. The area-balanced partitions using hMetis [26] and FLARE [27] algorithms, logic-vs-memory scheme, and Architecture-Aware scheme are compared. In the balanced-area and logic-vs-memory partitions, the chiplet areas are not sufficient to accommodate all the pins. In the Architecture-Aware partition scheme, the knowledge of architecture is used to come up with a partition in which the chiplets can accommodate all of their pins with reasonable pin-pitch. The Architecture-Aware partition scheme is used for the experimental studies presented in the latter part of this chapter. This scheme helps illustrate the Drop-in design approach, which allows several flavors of a 2.5D system with zero design cost. In this partition scheme, all core logic and 8KB of memory residing in the lower 8K address range are gathered into a Core-Chiplet. In the other Mem-Chiplet, only the rest of 8KB of memory is kept with a few control logic. As a result, the Core-Chiplet can operate as a standalone system with or without the Mem-Chiplet.

2.1.2 Technology Settings

TSMC 65nm technology is used to implement a 2D design and 2.5D chiplets. In this holistic flow, a unified environment is needed where both chiplet and package designs can be imported together

for holistic planning and extraction. Moreover, there is no publicly available PDK to design 2.5D packages for academic study. Therefore, the PDK is modified to create a unified chiplet-package co-design environment with all chiplet and package layers together.

Table 1: Technology parameters of the modified 65nm layer stack

Layer	Purpose	Width	Spacing	Thickness	Epsilon
M1-M7	Chip Internal Routing	TSMC	TSMC	TSMC	TSMC
ILD7	Inter-layer Dielectric	-	-	5 µm	2.0
M8	RDL1	10 µm	10 µm	5 µm	2.2
ILDR1	Inter-layer Dielectric	-	-	5 µm	2.0
M9	RDL2	10 µm	10 µm	5 µm	2.2
ILDR2	Inter-layer Dielectric	-	-	5 µm	2.0
M10	RDL3	10 µm	10 µm	5 µm	2.2
PP	Planar Passivation	-	-	1 µm	4.0
AP	Solder Pads	TSMC	TSMC	TSMC	TSMC

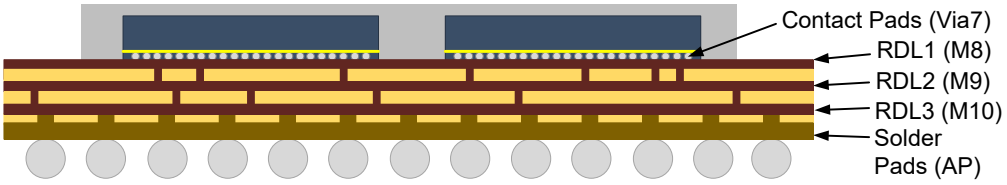


Figure 4: Package redistribution layer stack of the modified 65nm PDK

Table 1 shows the settings of the modified PDK. The lower seven metal layers (M1-M7) are used with their original settings for chiplet internal routing. The parameters of M8, M9, M10, and the relevant dielectric layers are modified to mimic the attributes of TSMCs InFO package routing layers. Though the most advanced InFO flavors can handle 0.8 µm/0.8 µm width/spacing [17], 10 µm/10 µm is used for a general setup. Fig 4 shows the layer stack of the modified PDK. For holistic extraction, this technology stack is characterized to generate an extraction-rule file. In an industrial design, this extraction-rule file would be provided by the packaging house through characterization of the chiplet-package technology combination they support.

2.1.3 Overall CAD Flow

The overall flow is illustrated in Fig. 5(b). When the RTL netlist is ready, the gate-level netlist is synthesized using a standard synthesis tool. The gate-level netlist is then fed to the partitioning

tool along with the partition scheme settings. The partition tool takes into account the impacts of package wires on chiplet partitions. Next, we prepate the top-level plan of chiplets and the package together in the same design environment set up with the unified PDK. We determine package floorplan and chiplet pin arrangement with an algorithm that reduces the package routing issues like long wires or detours and minimize package wire impacts on system performance. Next, we generate an initial package routing and estimate the package wireload on chiplet IO pins. We perform timing budget extraction of all chiplets and the package. Then, we split the overall design into individual chiplet and package sub-designs for parallel implementation. In Fig. 5(b), the "Chiplet Plan" boxes refer to plans of different chiplets, one plan for each chiplet. Though these chiplet plans are related through the top-level constraints, each plan is independent and all plans can be implemented in parallel.

After co-planning and RDL routing, chiplets and the package can be implemented independently with contexts and constraints propagated from the top level. Package design is performed utilizing the chiplet footprints, their connectivity, and the timing budget of package wires. The physical design of each chiplet is similar to the traditional 2D chip design flow with some additional constraints imposed by the top-level plan. After placement and routing, Design Rule Checking (DRC) is performed on all chiplets individually. If all chiplets pass the DRC, the entire system is assembled together for further optimizations and analyses.

The Design Assemble step combines chiplet and package designs into the same unified design environment as in the top-level planning stage. Because of this, optimization and analysis can capture chiplet-package interactions and perform adjustments to improve system performance and reliability. Holistic extraction is performed and the result is used for STA and timing context generation. These timing contexts are used to perform the next iterations of individual chiplets. This holistic optimization using standard tools improves system performance with buffer resizing, time borrowing, re-routing, etc. Following iterations can be carried out if there is a scope of improvement, but with a good estimation at the beginning, the second iteration is generally accurate enough. Finally, all the finished designs are assembled for full-system extraction,

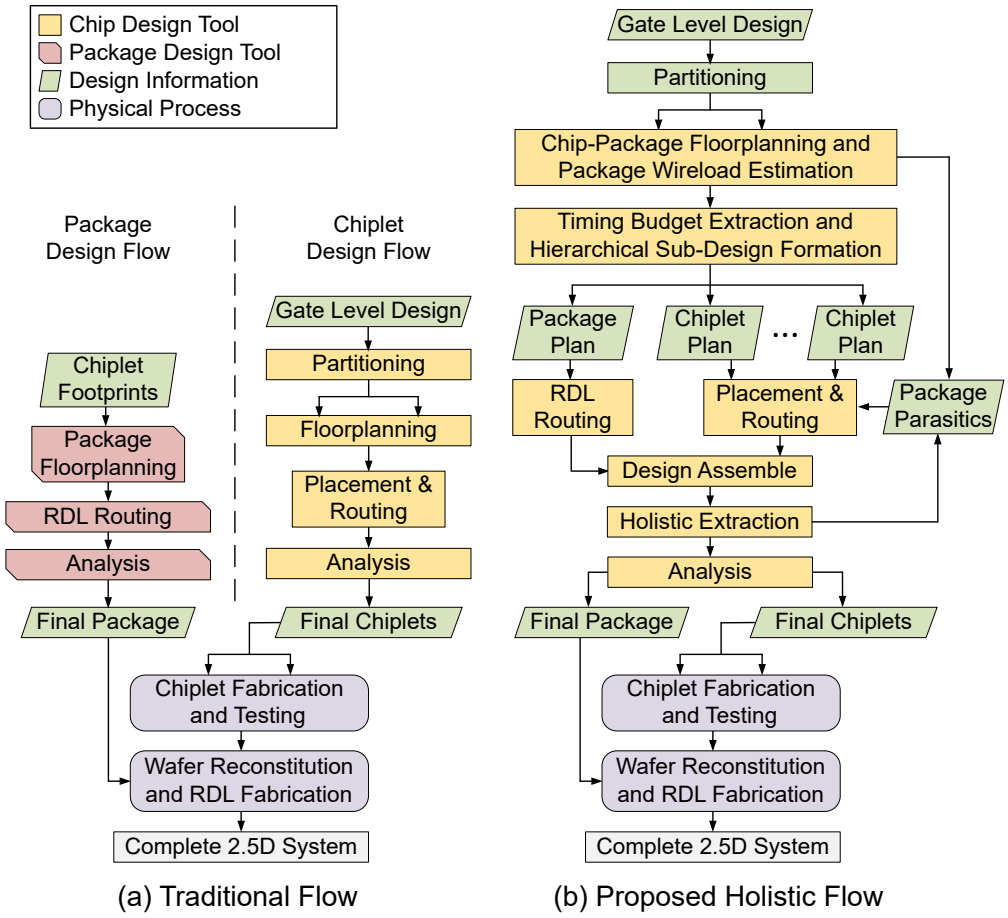


Figure 5: The traditional Die-by-Die (DbD) design flow of a 2.5D system versus the holistic iterative optimization flow

analysis, and sign-off verifications.

2.2 Chiplet-Package Co-Planning and Modeling

In 2.5D systems, RDL and package planning is critical to minimize inter-chiplet signal delays due to package wires. Otherwise, even though the chiplets may obtain very high performance individually, the overall system will perform poorly due to timing bottlenecks through package wires. As the package routing is highly dependent on chiplet pin configurations, they need to be planned together. At this step of the flow, the package floorplan, RDL routing, and chiplet pin configurations are optimized in a holistic way to minimize the impact of package wire delay on the overall system performance.

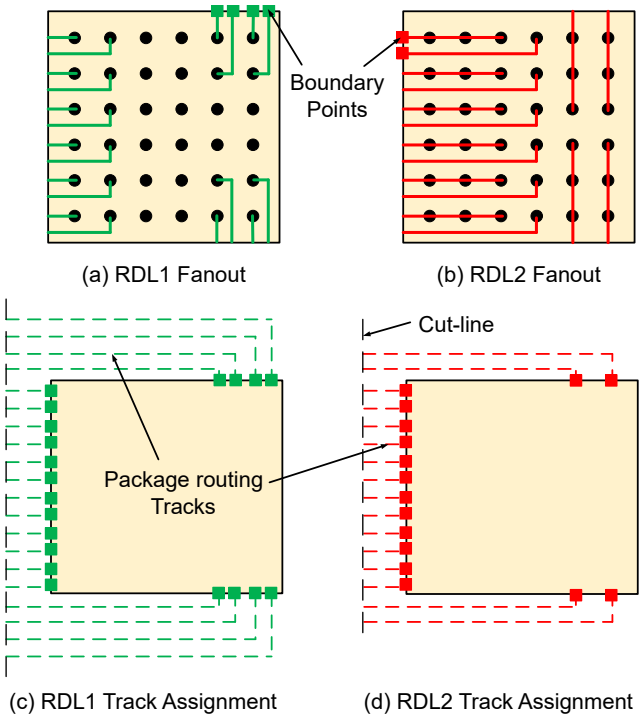


Figure 6: Illustration of pin fan-out and track assignment of a chiplet with 6×6 pin grid and two RDLs.

2.2.1 Top-Level Planning

The RDL routing problem of a 2.5D package is different from the conventional chip routing problem. Existing works [28, 20, 19] try to solve the routability between chiplet pins in the system. However, compared to the chip routing problem, the number of nets on the package level is much fewer, and signal integrity issues are mainly caused by skewed long wires. As a result, minimizing total wire-length is not always the primary concern. Several other factors like bus delay skew, signal-integrity, the inductive effect of long package wires, EMI effects, etc. can play a critical role. All these factors can be considered in the top-level planning stage of a 2.5D system. A strategy is presented, which focuses on developing a compact RDL routing plan with short and uniform wire-lengths to minimize routing issues like congestion, detours, and unequal bus wire delays between chiplets.

Chiplet dimensions and pin pitch are determined based on the chiplet area and pin count. In the experimental implementation, the Core-Chiplet has dimensions 520 µm×475 µm and a total

of 100 pins. The Mem-Chiplet has dimensions 415 µm×230 µm and a total of 60 pins. The pins of the Core-Chiplet are arranged in a 10×10 grid and those of the Mem-Chiplet are arranged in a 6×10 grid. In both chiplets, the pin pitch is 40 µm in both directions of the grid. Without loss of generality, two chiplets are considered at a time in the co-planning step. In this strategy, signals are assigned to chiplet pins after the package floorplan and routing are determined. The top-level package and chiplet plans are determined through pin fan-out of chiplets, RDL track assignment, package floorplanning and routing, slack-based greedy signal assignment of package wires, and package wireload estimation. Algorithm 1 describes our co-planning strategy.

Algorithm 1: RDL Planning Algorithm

```
1  Calculate area required for the chiplets
2  Generate pin array based on pin pitch and chiplet area
3  sideOrder = [near cut-line, top, bottom, opposite side]
4  layerOrder = [RDL layers from bottom to top]
5  foreach Chiplet do
6      foreach s in sideOrder do
7          foreach l in layerOrder do
8              Route pins to the Boundary Points of s on l
9      Sort BoundaryPoints in increasing order of their distance from cut-line
10     foreach bp in BoundaryPoints do
11         Assign the nearest available track to bp
12 while Floorplan not valid do
13     Floorplan = New relative position of the chiplets
14     Check if Floorplan is valid
15 Connect pin pairs routed to the same track
16 AssignSignals(Tracks, Nets, Slack)
17 WireLoadEst(Tracks, PDK.WireLoadModel)
18 Generate TCL script and SDC files
```

2.2.2 Pin Fan-Out and RDL Track Assignment

Before a chiplet pin can be routed externally, it needs to cross its chiplet boundary. A greedy strategy is used to fan out and track the chiplet pin assignment, which tries to use short and straight wires within a minimum number of package layers. In Algorithm 1, lines 5-11 show the pin fan-out and track assignment strategy. For the sake of illustration and explanation, a cut-line is

assumed between two chiplets, as shown in Fig. 7. This cut-line acts as the routing target in this step. As many internal pins as possible are brought to the chiplet boundary using all the RDL routing tracks that cross the boundary. The boundary locations where the pins are routed to are named as "Boundary Points." This is performed in a specific order on all sides of the chiplet. For each side, the layer touching the contact pads is routed first, followed by the subsequent RDLs. From line 3 of Algorithm 1, the side order is determined based on their rough distance from the cut-line. The number of rows/columns of pins that can be routed to boundary points depends on the pin pitch in terms of the package routing track. As shown in Fig. 6, if the pin pitch is two tracks, two rows/columns of pins adjacent to that side can be routed to the Boundary Points following those tracks.

Next, tracks are assigned to these Boundary Points. Boundary Point closest to the cut-line is assigned with its nearest track first. From line 9 of Algorithm 1, Boundary Points are sorted based on their distances from the cut-line. As a result, in the track assignment queue, the Boundary Points facing the cut-line comes first, followed by Boundary Points on the perpendicular side and the opposite side, sorted in increasing order by their distances to the cut-line. The opposite side is least preferred because of the detours introduced to reach the cut-line. Fig. 6 shows the pin fan-out and track assignment of a chiplet with a 6×6 pin grid and two package layers.

2.2.3 Package Floorplan and Routing

Based on the track assignment of chiplets, their relative locations are determined. These relative locations will determine the package floorplans, chiplet connectivity, and RDL routing. In the current strategy, a relative position is accepted between the chiplets that can produce sufficient overlap of the tracks to allow all their connections. Lines 12-14 of Algorithm 1 describe the strategy to determine the package floorplan.

Fig. 7 illustrates this strategy for connecting four pins between the chiplets using only one RDL, where the dashed white lines show available tracks crossing the cut-line. The thick lines connected to chiplets represent assigned tracks to chiplet pins. The track assignment strategy

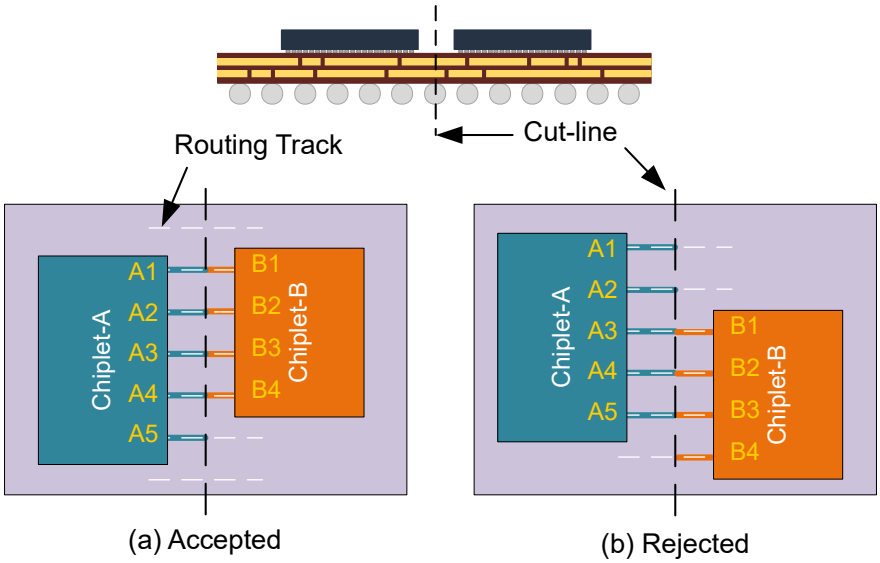


Figure 7: Illustration of the floorplanning strategy: (a) A selected solution that satisfies the pin connectivity requirement, (b) a rejected floorplan while finding the relative location

routes the pins of Chiplet-A and Chiplet-B to their nearest tracks crossing the cut-line separately. Next, while exploring different possible relative positions between chiplets, the floorplans similar to Fig. 7(b) are rejected as those have insufficient track overlap for four connections. Among two viable solutions, in this case, the floorplan in Fig. 7(a) is chosen arbitrarily, which supports the number of connections between the chiplets. After finding the relative position, the connectivity among the pins of the two chiplets is defined, which are routed to the same track crossing the cut-line. In this example, pin A1 of Chiplet-A to pin B1 of Chiplet-B are connected because they are routed to the same track. Similarly, pins A2, A3, and A4 of Chiplet-A will be connected to pins B2, B3, and B4 of Chiplet-B, respectively. Unconnected pins of the chiplets, like pin A5 of Chiplet-A, can be used to connect with some other chiplets or to act as external I/O.

2.2.4 Signal Assignment

With the connectivity defined, both chiplet floorplan and pin assignment can be prepared in compliance with the rest of the package plan. One way to perform the signal assignment would be based on chiplet floorplans. In this strategy, a designer can prepare some initial floorplans and assign signals to the pins. Another way would be to determine the signal assignment of the chiplet

pins based on timing requirements and then adjust chiplet floorplans to best-suite the pin configurations. The latter approach is followed here.and a greedy algorithm is applied for the signal assignment. The AssignSignals() function in Algorithm 2 describes the signal assignment strategy. Performing STA analysis on the synthesized gate-level netlist, timing slacks of all package wires are collected. Based on the floorplan and routing obtained in the previous steps, all track lengths connecting chiplet pins are calculated. As shown in lines 4-5, tracks and nets are sorted by their lengths and slacks, respectively. As a result, in lines 7-9, the net with the smallest slack is assigned to the track with the shortest length. This greedy strategy assigns timing-critical nets to shorter package wires and thus minimizes the package-wire delay overhead on them. This eventually improves overall system performance.

2.2.5 Package Wireload Estimation

When RDL routing and signal assignment are complete, parasitic loads at chiplet IOs due to package wires can be estimated. Being aware of the output load, during the optimization steps, chiplet design tools can make necessary adjustments like buffer insertion, cell resizing on IO nets. At this point, the goal is to perform a rough estimation of package wire loads to complete the first iteration of chiplet implementation. More accurate parasitics can be extracted from the assembled design for the second iteration of chiplet implementation. The package wireload is calculated as a linear function of the wirelength. The function WireLoadEst() in Algorithm 2 describes the wireload estimation method. A wireload model is a list of values that represent the capacitance per unit length of package wires. These values are calculated from technology settings and package wire dimensions.

At the end of the co-design steps, as depicted in line 18 of Algorithm 1, the RDL planner tool generates a TCL script to implement the package routing and SDC files for all chiplets specifying wireload on IO pins. In the SDC file, the capacitance estimated by the WireLoadEst() function is specified as the wireload of the corresponding pin. With this strategy, the RDL planner can directly handle one-to-one pin connections between two chiplets. The point-to-point connection

is prioritize here since this is the most commonly used connection type on the package level. A multi-point connection can be handled by breaking it down into multiple point-to-point connections and then applying our strategy. Multiple chiplets can be handled by grouping the chiplets which are already interconnected into a single chiplet-like entity and perform routing between the group and another chiplet.

Algorithm 2: Signal Assignment & Wireload Estimate

```
1  Function AssignSignals(Tracks, Nets, Slack):
2      foreach track in Tracks do
3          track.length = calc_path_len(track.path)
4      sorted_tracks = sort_by_length(Tracks)
5      sorted_nets = sort_by_slack(Nets, Slack)
6      set next_track = 0
7      foreach net in sorted_nets do
8          sorted_tracks[next_track].signal = net
9          next_track += 1
10     return
11
12 Function WireLoadEst(Tracks, WireLoadModel):
13     foreach track in Tracks do
14         cap_per_len = WireLoadModel[track.layer]
15         track.load = track.length×cap_per_len
16     return
```

2.3 Physical Design

The physical design of both chiplets and the package can be implemented using any commercial chip design environment that supports hierarchical design flow. The Cadence Innovus is used to perform the hierarchical implementation of the package and chiplets in the experimental 2.5D system. The design environment is set up with the modified TSMC 65nm PDK and the entire system is loaded into the environment. Chiplets appear as modules in this design environment. Based on the plan generated by the RDL planner, the chiplets are placed on the package and their signal assignments are defined. Using the scripts generated by the RDL planner, the chiplet pins are routed on RDLs. Then the timing budget of chiplets and the package are extracted. After this

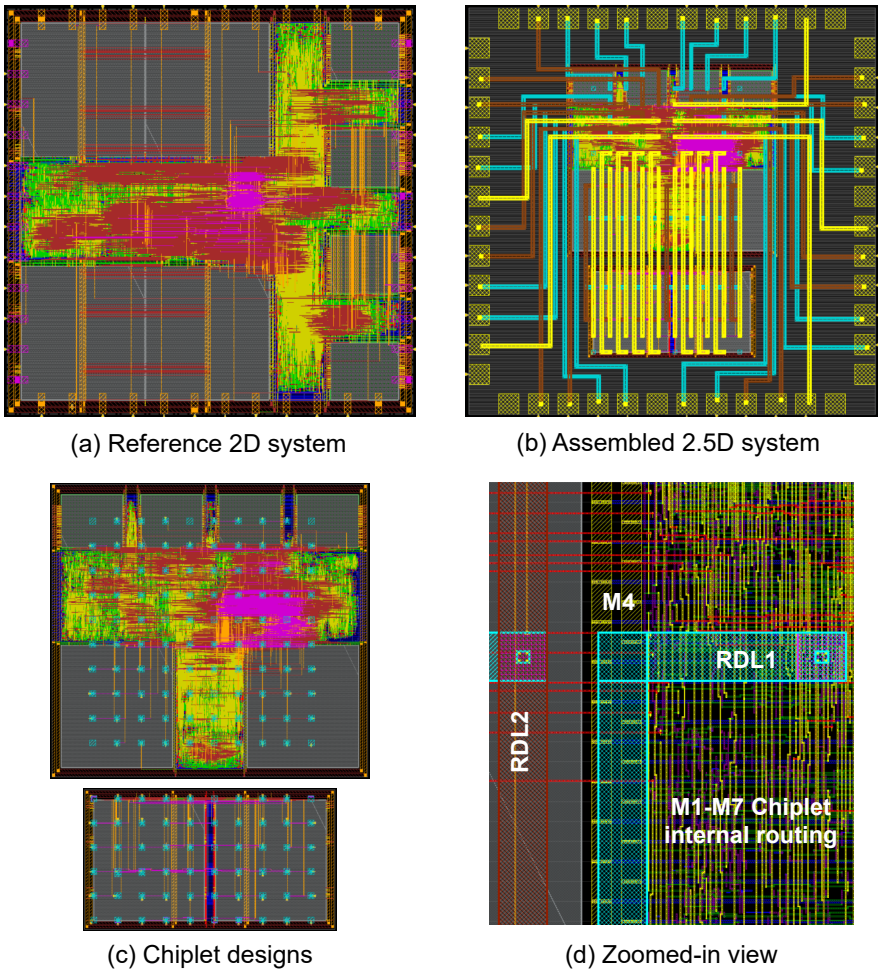


Figure 8: Design layouts of (a) reference 2D system, (b) assembled 2.5D system with chiplet and package layers, (c) designs of Core-Chiplet (top) and Mem-Chiplet (bottom), and (d) Zoomed-in view of the assembled design.

step,chiplets and the package are separated as hierarchical sub-designs, and can be implemented in parallel in their own design environments.

2.3.1 Hierarchical Implementation

During implementation, each chiplet is treated as a single 2D design with some extra constraints imposed by the top-level plan and designed using traditional chip design techniques. The initial SDC file, which defines the chiplet context (like IO delay, output load, etc.), is modified to include the wireload on chiplet pins estimated by the RDL planner tool. In the top-level planning stage, initial chiplet floorplans are prepared. This floorplan can be adjusted if necessary without

changing the pin configuration specified by the top-level plan. After fixing the floorplan, the Power Distribution Network is designed to ensure reliable power delivery to standard cells and memory macros. Standard tools are used for standard cell placement, clock network design, routing, and post-routing optimizations. Finally, filler cells and metal fills are used to fulfill the density requirement. Fig. 8(c) top sub-figure shows the Core-Chiplet, which contains all logic blocks and 8KB memory in the lower address range. Fig. 8(c) bottom sub-figure shows the Mem-Chiplet, which contains the other 8KB memory in the upper address range.

Package design can be implemented in parallel with chiplet designs. However, more accurate and reliable optimization of the package can be performed if interface timing models of chiplets extracted after their implementation are used. The RDL planner generates routing scripts for inter-chiplet routing at the end of the co-planning step. These scripts are utilized to finish inter-chiplet routing. Based on the package floorplan and inter-chiplet routing, package external IOs are placed. Chiplet pins that are not used in inter-chiplet connectivity are assigned as external connections to package IO pads. Fig. 8(b) shows the package design of the 2.5D system that integrates the chiplets shown in Fig. 8(c).

With routed chiplets and package layouts, they are imported into the integrated design environment again. To ensure manufacturability, DRC is performed on each of the chiplet and the package before design assembly. Fig. 8(d) shows a zoomed-in view of the assembled design, which shows traces from both chiplets and the package in the unified environment. Holistic extraction is performed on this assembled design using the extraction-rule file characterized for the chiplet-package unified technology. As all chiplets and the package are combined together in the same environment, all interactions between chiplets and the package are captured accurately in the extraction.

2.3.2 Holistic Extraction

Holistic extraction can be performed using any commercial extraction tool that supports hierarchical extraction flow. The Synopsys StarRC LEF/DEF flow is used to perform the holistic

Table 2: Holistic capacitance (in fF) extraction results

Metal Layer	M1-M5	M6	M7	RDL1	RDL2	RDL3
M1-M5	16348	222.5	446.7	185.3	18.61	10.18
M6	222.5	137.1	32.81	51.7	4.168	2.149
M7	446.7	32.81	371.1	32.43	1.459	1.891
RDL1	185.3	51.70	32.43	65.67	399.3	11.19
RDL2	18.61	4.168	1.459	399.3	103.3	390.5
RDL3	10.18	2.149	1.891	11.19	390.5	115.3
Ground Capacitance						
Metal Layer	M1-M5	M6	M7	RDL1	RDL2	RDL3
Capacitance	31842	1526	477	853	251	420

extraction. Table 2 shows the coupling capacitance extraction result of the final design. For readability, capacitance numbers from M1 to M5 are merged. In the traditional die-by-die approach, it is not possible to accurately capture the interactions between chiplet and package routing layers. In the holistic method, these interactions are captured which are presented in the last three columns of the table, RDL1, RDL2, and RDL3. If we notice, there exists sufficient coupling between RDL1 of the package and M6-M7 of chiplets. Moreover, it is evident from the numbers that the coupling of RDL1 with M6 is greater than that with M7. As M7 is the topmost chiplet routing layer, it is expected that the coupling between RDL1 and M7 should be greater. However, as the routing on M6 is significantly greater than that on M7, and routing tracks of RDL1 and M6 run in the parallel direction, the effective overlap between RDL1 and M6 is much greater than that with M7. This detailed interaction can only be captured in a holistic extraction method. The extraction result can then be utilized to incrementally improve the system performance, signal integrity, and system reliability. In the next section, a set of design case studies is presented that reveals the impact of chiplet-package interactions and how the holistic extraction result can be utilized to iteratively improve the design.

Table 3 shows a comparative study between die-by-die and holistic extraction results. The die-by-die extraction result is calculated by performing extractions on individual chiplets and the package separately and then adding capacitance values of corresponding layers. As seen from Table 3, die-by-die extraction severely over-estimates the ground capacitance, especially on

Table 3: Comparison of Holistic vs. Die-by-Die ground (GCAP) and coupling (CCAP) capacitance extraction results (in fF)

Metal Layer	M1-M5	M6	M7	R1	R2	R3
DbD GCAP	31973	1568	485	1285	323	433
Holi GCAP	31842	1526	477	853	251	420
DbD GCAP Err	0.41%	2.77%	1.60%	50.77%	28.8%	3.19%
DbD CCAP	23366	394	864	547	830	517
Holi CCAP	23732	450	886	746	917	531
DbD CCAP Err	-1.54%	-12.5%	-2.57%	-26.5%	-9.55%	-2.75%

package layers. This over-estimation is due to the absence of chiplet routing layers between the package layers and the reference ground plane. More alarming errors are observed in coupling-capacitance. Die-by-die extraction severely underestimates the coupling capacitance on all layers as it cannot capture the interactions between chiplets and the package layers. This large error in parasitic extraction can cause severe signal integrity issues leading to system failure. Therefore, holistic consideration of chiplet and package interactions is a must in high-density 2.5D packaging technologies.

One limitation of holistic extraction is that the existing commercial extraction tools cannot perform holistic extraction when heterogeneous technologies are involved. However, this limitation is not inherent to the holistic flow. This limitation can be overcome by extending the extraction tools to handle multiple heterogeneous technologies. An intermediate solution is to perform in-context parasitic extraction per technology and stitch them together carefully to create the holistic extraction result.

2.3.3 Iterative Optimizations

After design assembly and analyses, if the target performance is not achieved and discrepancies between estimated package parasitics with extraction results are observed, iterative implementation of chiplets can be conducted. If active packaging material is used, a similar optimization procedure can be performed on the package layer as well. In the first iteration of the chiplet design, the package wireload is a rough estimation based on package wirelength. Thus almost always some room for improvement can be expected. After design assembly and holistic

extraction, STA analysis is performed on the design with the holistic extraction result. Based on this analysis, new timing contexts are created for all the chiplets. In the STA analysis, timing paths through the package are modeled, including the cells within the driver and receiver chiplets. This makes the cross-boundary co-optimization between chiplets possible. One limitation of existing STA tools is that they only consider resistive and capacitive parasitic elements of the nets. However, package wires exhibit significant inductive behavior. Though in this paper, we only consider the capacitive impact of the package wires, this same methodology can be applied to consider other elements of the package wire, which affect the overall system performance.

As accurate parasitic information is available through holistic extraction, it is possible to generate a tighter timing budget for the next iteration. The Synopsys PrimeTime is used to create the updated timing contexts utilizing its context characterization feature. The updated timing context is exported as an SDC file for each chiplet. This SDC file contains all the details of the timing contexts of each IO pin of a chiplet. For output pins, it specifies maximum transition time, wireload, pin-load, and output delay. For input pins, it specifies minimum/maximum allowed capacitance, maximum fanout, driving cell, and input delay. For all the delay information, clock latency is also specified. Using these updated timing contexts, all chiplets are reimplemented and adjusted for the package overhead. These timing contexts can be used to perform iterative optimization of the package design as well. There can be several iterations of assembly, extraction, timing context creation, and reimplementation until it is no longer possible to improve the system performance or the target performance is met. However, with a good estimation in the first iteration, a second iteration is generally good enough to meet the best system performance.

2.4 Two-way Partition Design Study

Several design cases are prepared to study the impact of chiplet-package interactions on the system. It is found that holistic extraction results can be utilized to significantly improve system performance. In this section, some of these designs and analysis results are presented. Since this is the first time a holistic chiplet-package design methodology is performed, not every detailed

analysis step is performed, such as power integrity and signal integrity, thermal analyses, etc. However, since the holistic design flow is compatible with all modern chip and package design CAD tools, it is likely that those analyses can be easily added with a few minor modifications.

2.4.1 Design Case Variants

Case-1: Reference 2D Design: A 2D design of the microcontroller system is implemented as a reference design using TSMC 65nm technology with lower seven metal layers. The gate-level netlist obtained after synthesis and before preparing chiplet partitions is used in this design. After trying out several floorplans, a square floorplan was selected with a side length of 600µm, as shown in Fig. 8(a). PDN, cell placement, clock network design, routing, and post-routing optimizations are performed using standard chip design tools. The finished design achieves 400 MHz maximum system frequency. Table 4 Case-1 column shows the parameters of the finished design.

Case-2: Context-Free 2.5D Design: This case is a context-free single-pass design that resembles the traditional die-by-die approach. Chiplets and the package are designed independently without using the context creation step as in the holistic flow. Though the RDL planner generates the top-level plan, it does not perform package wireload estimation. However, design assembly and holistic extraction is performed to capture chiplet-package interactions. This design case reveals the impact of the package on chiplets and the consequent degradation of the overall system performance. Table 4 Case-2 column shows the parameters of this design case.

Case-3: Context-Aware Optimized 2.5D Designs: This case is designed in the holistic flow and optimized using iterative context creation and reimplementation of chiplets, as discussed in Section 2.3.3. Chiplet-package interactions are included as much as possible in the design and optimization steps. As discussed previously, the RDL planner prepares the top-level plan and calculates package wireload estimation, which is used in the first iteration of chiplets implementation. After design assembly and holistic extraction, extracted parasitics are used to

perform STA and create chiplet timing contexts for the next iteration of chiplets implementation. The last two columns of Table 4 show the parameters of two different iterations of this design.

2.4.2 Holistic Analysis and Optimization

Case-1 2D implementation is considered as the reference design. Due to the inter-chiplet RDL wire overhead, it is expected that 2.5D implementations will have worse performance. In the Case-2 design, which resembles the die-by-die design approach, after applying all possible traditional optimizations, all chiplets achieve 400 MHz operating frequency, the same as the 2D design. However, the overall 2.5D system can only run at a maximum frequency of 366 MHz. The slowest paths are between the chiplets through the package, resulting in a slower clock frequency. This result reveals that the holistic extraction method can capture the package impact on the overall system performance. This package overhead is overlooked in the die-by-die design approach. As a result, the die-by-die analysis will report an inaccurate system frequency. The holistic extraction and analysis flow can accurately capture the package overhead on the system performance and report the frequency at which the system can run reliably.

In the first iteration of the Case-3 design, a predictive package wireload model is used in chiplet implementation. Though it is a very rough estimate based on a linear model, this design achieves an operating frequency of 384 MHz. Compared to the performance gap of 34 MHz between the 2D implementation and the Case-1 2.5D implementation, this is an approximately 50% reduction in the performance gap. This result reveals the importance of considering chiplet-package interactions, even in the early planning stage.

In the second iteration of the Case-2 design, timing contexts created using holistic extraction results are imported during chiplet implementation. These contexts have an accurate picture of the overall system. Using these contexts, the chiplet design tools can adjust chiplet designs to compensate for the delay introduced by package wires. As a result, in the second iteration, the 2.5D system achieves a 395 MHz operating frequency, which is very close to the 2D system performance. As the critical path is from the Core-Chiplet to the Mem-Chiplet through the

Table 4: Analysis result comparison of the microcontroller system

Design Case	Case-1	Case-2		Case-3 first iteration		Case-3 2nd/final iteration	
Chip Design	2D Chip	Core Chiplet	Ext. Mem Chiplet	Core Chiplet	Ext. Mem Chiplet	Core Chiplet	Ext. Mem Chiplet
Logic Gates#	24141	23933	20	23918	15	23909	0
Buffer/Inverter#	4760	4684	20	4634	15	4653	0
Die Size (um $\times$ um)	600 $\times$ 600	520 $\times$ 475	415 $\times$ 230	520 $\times$ 475	415 $\times$ 230	520 $\times$ 475	415 $\times$ 230
Total Chip Wirelength (mm)	551.974	495.578	14.289	488.11	13.842	485.923	12.373
M6 Wirelength (mm)	15.128	12.978	2.847	13.607	4.052	11.866	4.579
M7 Wirelength (mm)	8.562	19.084	1.991	18.117	2.312	17.446	3.264
Max Frequency	400 MHz	366 MHz		384 MHz		395 MHz	
Performance Gap	0%	100%		47.05%		14.70%	
Chip Power	20.1 mW	18.4 mW	2.504 mW	18.2 mW	2.506 mW	18.2 mW	2.576 mW

address bus, the size and number of buffers in the Core-Chiplet increased, while the redundant buffers in the Mem-Chiplet are removed. All these optimizations are performed by the chip design tools without any special setting other than the timing contexts created using the holistic extraction result. More iterations are performed afterward, but there is no significant improvement in system performance. That is why the second iteration is taken as the final design of Case-3. This result reveals that with proper considerations of the chiplet-package interactions, it is possible to reduce the inter-chiplet overhead and optimize the overall 2.5D system performance. In this design case, the performance gap between the 2D implementation and the Case-2 2.5D system is reduced by 85% through the holistic extraction and iterative optimization flow.

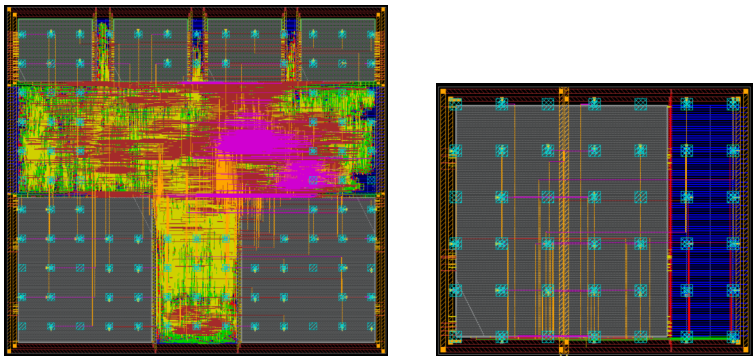
2.5 Agile Multiway Design Techniques

Though the holistic design flow and planning strategy are illustrated based on a two-way partition design, it can be easily extended for multiway partitioned designs. To illustrate a multiway partitioned system, the application of novel design techniques enabled by 2.5D integration, and the design flexibilities offered by the flow, a three-way partition implementation of the microcontroller system is presented here.

2.5.1 Three-way Partition Design

In this implementation, the 8KB Mem-Chiplet of the previous 2.5D system is further divided into two 4KB Mem-Chiplets. Fig. 9 shows the chiplets for this three-way partition design. This way, now the 2.5D system can have three different flavors with 16KB, 12KB, and 8KB memory capacities. Fig. 10 shows all these flavors of the system. Fig. 10(b) shows the system with all three chiplets with 16KB memory. The RDL plan of this design is prepared in two stages. In the first stage, only the two 8KB Mem-Chiplets are considered. These two chiplets share 12 connections on the address bus. The RDL planning tool routes these nets using straight horizontal wires on RDL1. These wires can be seen in Fig. 10(b) as horizontal blue wires in the lower half of the package. In the second stage of RDL planning, these two chiplets are considered as a single

chiplet-like group. For the RDL router, dummy pin locations are specified on the horizontal RDL1 wires between the two chiplets. The RDL planning tool routes the connections between the Core-Chiplet and this combined chiplet-like group to finish the inter-chiplet routing. The address bus is routed on RDL2 and RDL3 and form T-connections with the RDL1 wires between the Mem-Chiplets. Finally, the remaining pin locations of the Core-Chiplet are used as external IOs. With this top-level RDL plan, chiplets are implemented following the iterative optimization flow. As seen from the first row of Table 5, the optimized 16KB system achieves a maximum operating frequency of 380 MHz.



(a) Core-Chiplet (b) 4KB Mem-Chiplet

Figure 9: Layouts of the chiplets for the three-way partition design study

2.5.2 Drop-In Approach

In Fan-out Wafer Level Packaging (FOWLP), a reconstituted wafer is built using the Known-Good-Dies (KGD) of the chiplets. In this step, a chiplet can be deliberately left out from the reconstituted wafer to design low-cost flavors of a system with limited capabilities while keeping all optical masks untouched. Let us call this the “Drop-In” design approach. This name reflect the fact that the whole system could be created if the missing chiplet is dropped into the package. In the three-way partitioned design, a 12KB system can be designed if the second Mem-Chiplet is excluded from the package. This design approach requires zero design costs but offers the end-users to choose from several options as per their requirements. The holistic flow can handle this design approach. Fig. 10(c) shows the drop-in design with 12KB memory. The

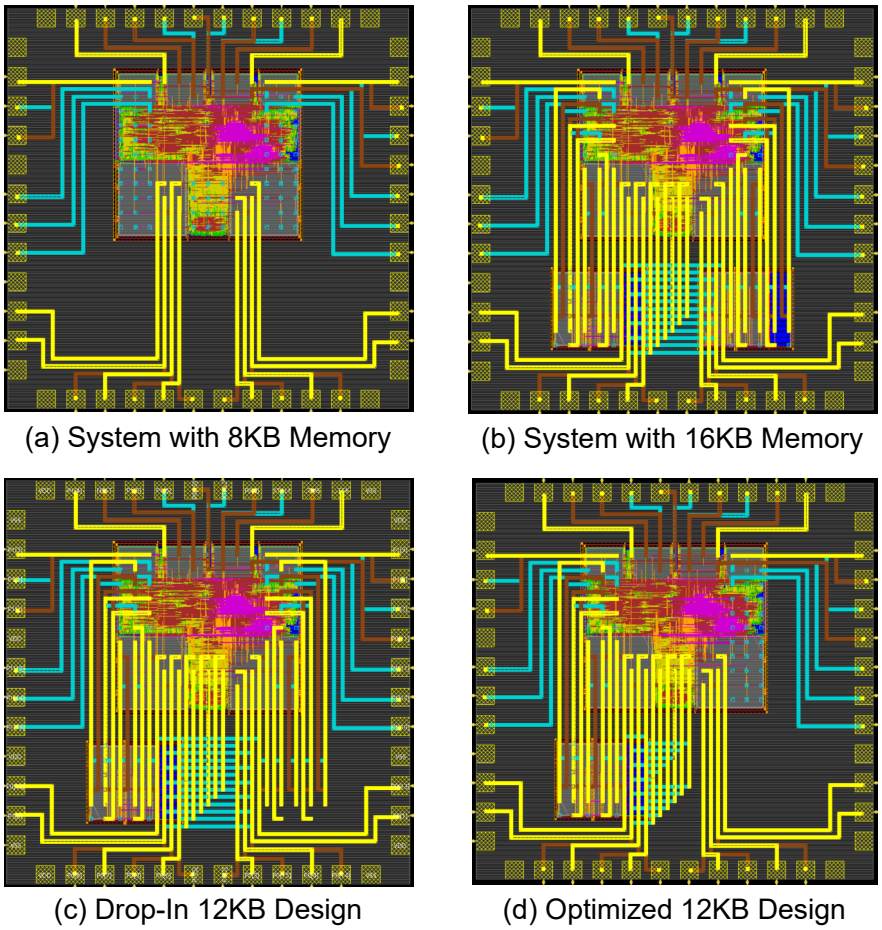


Figure 10: Design layouts of (a) Core-only system with 8KB memory, (b) optimized full system with 16KB memory, (b) 12KB design using the Drop-in approach, and (d) Optimized 12KB design using Pay-as-You-Use approach.

second Mem-Chiplet is excluded from the package, and holistic extraction and analysis are performed after design assembly. The second row of Table 5 shows that, in the absence of the second Mem-Chiplet, the system performance improved from 380 MHz to 390 MHz without any further optimizations on the chiplets.

2.5.3 Pay-as-You-Use Approach

This is another approach, similar to the Drop-In approach, to develop several flavors of a 2.5D system. In this approach, each design flavor is customized depending on the usage of systems components. The penalty paid in the system performance and power depends on the use of the system resources. Fig. 10(d) shows a 12KB implementation of the three-way partitioned system

designed in this approach.

Unlike the Drop-In design, the package routing is modified to remove redundant package wires related to the second Mem-Chiplet. Another incremental iteration of the holistic optimization flow is performed, which automatically adjusts package wire drivers of the chiplets according to design needs. Since all steps are performed in a standard ASIC design environment, this process is fully automated and takes less than an hour, enabling agile design customization. As seen from the third row of Table 5, the system performance improved from 390 MHz of the Drop-In design to 396 MHz with the reimplementation. Though in this design case, it is not a huge performance gain, this illustrates the flexibility and optimization capability of the flow, which can be utilized by system designers to quickly generate customized flavors of 2.5D systems and reduce the turn-around time.

Table 5: Comparison of three-way partition design cases

Design Flavor	LPD (ns)	Frequency (MHz)	Power (mW)	RDL Wirelength (μm)
16KB Optimized	2.62	380	19.7	46826
12KB Drop-in	2.56	390	18.8	46826
12KB Optimized	2.52	396	18.8	35541
8KB Core-only	2.50	400	18.1	20905

Of course, another flavor of the system can be designed by removing both the Mem-Chiplets and keeping the Core-Chiplet only in the package. This system is shown in Fig. 10(a). Holistic extraction and analysis are performed on this system. This system achieves an operating frequency of 400 MHz, the same as that of the reference 2D design, demonstrating no observable delay overhead introduced by package wires.

2.6 Silicon Validation with Tape-Out

To validate the holistic flow in silicon, a shared-block design containing a 2D system and a 2.5D implementation of the microcontroller is taped out . TSMC 65nm PDK is used as the implementation technology. The top two routing layers of the chip design PDK are modified to be used as RDLs. As this chip is designed to be manufactured through mimicking the attributes of

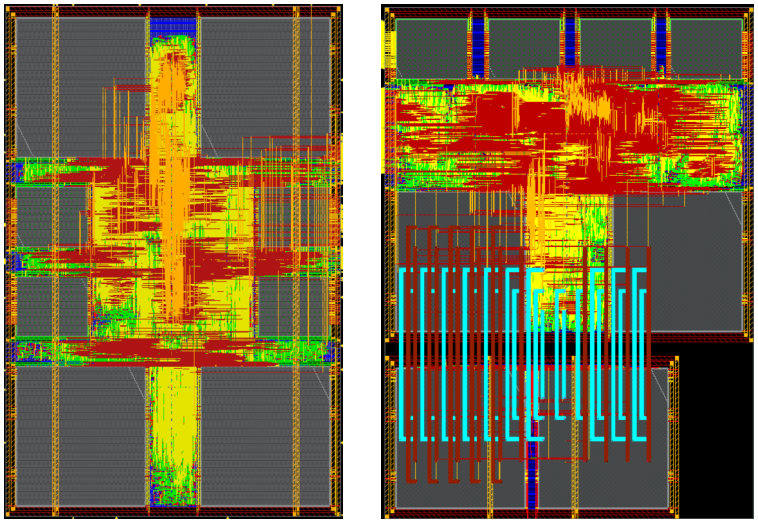
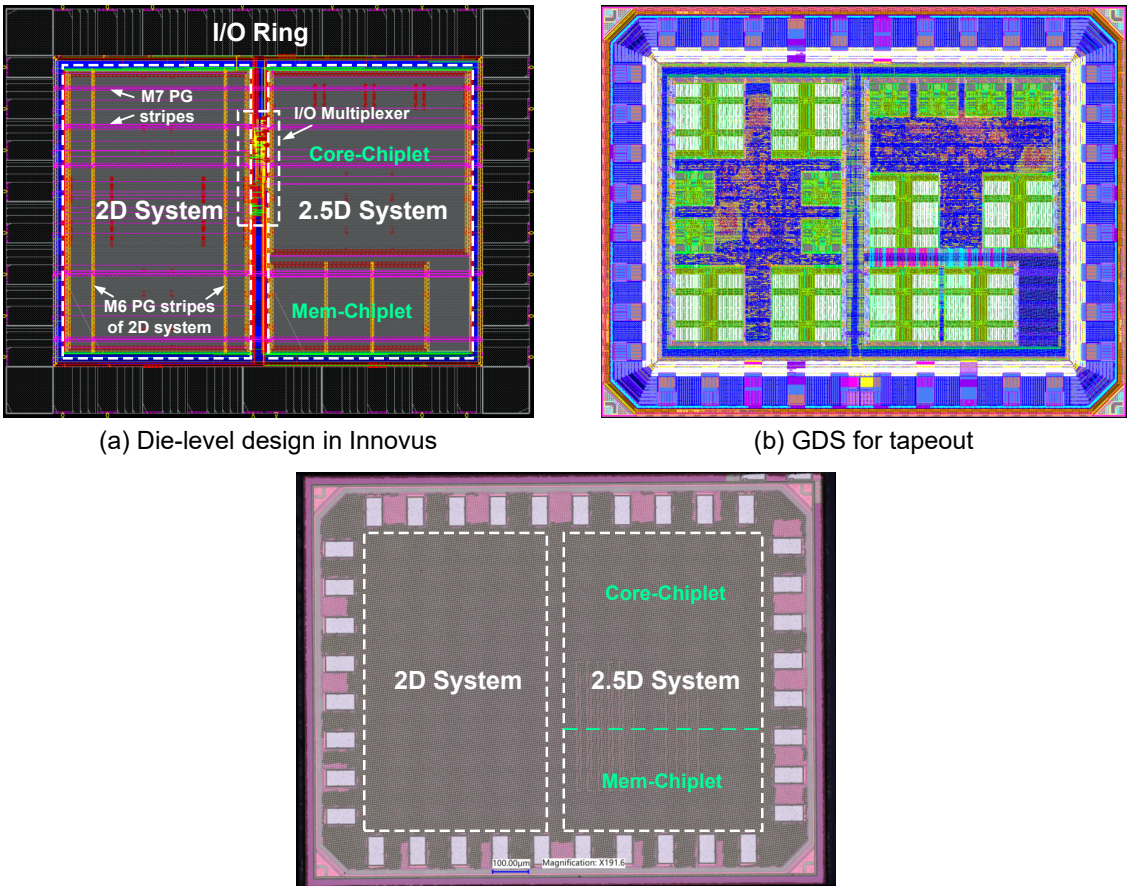


Figure 11: System designs for tape-out: (a) Reference 2D system, (b) assembled 2.5D system.

2.5D RDLs, several design considerations are made. The system architecture of the ARM Cortex-M0 microcontroller is originally designed to be implemented as a System-on-Chip (SoC) at a target frequency of 100 MHz in TSMC 65nm technology. For this mimicked technology stack, there is no foundry-provided extraction-rule file. Moreover, for the shared-IO design and chip testing using simple logic analyzers, several testing logic are embedded within the chip. For all these reasons, though the target frequency is 400 MHz, the taped-out system runs at around 100 MHz.

2.6.1 2D and 2.5D System Designs

A reference 2D system is designed along with a 2.5D system for tape-out. Lower six metal layers are used for designing the 2D system and for performing the internal routing of the 2.5D chiplets. The 2D system is designed using traditional chip design flow. The chiplets of the 2.5D system are designed in the holistic design and optimization flow. In both systems, M5 and M6 are used for designing the PG ring around the core. The PG rails on M1 and PG stripes on M6 supply power to the standard cells and macros, respectively. Because of the shared-block design approach, only the inter-chiplet connections in the 2.5D system are routed using RDLs. The external IOs of both systems are placed on M7, next to an IO multiplexing module. Fig. 11(a) shows the finished 2D



(a) Die-level design in Innovus

(b) GDS for tapeout

(c) Microscopic die-shot

Figure 12: Final design for tape-out and the fabricated die: (a) Die-level design, (b) combined GDS for tape-out, (c) microscopic image of the taped-out die.

system. Fig. 11(b) shows the assembled 2.5D system.

2.6.2 Shared IO Design

To save the pin area, a shared-block approach is used in the tape-out design. This also helped satisfy the minimum area requirements of the foundry while saving the IO cell area. The 2.5D implementation is combined with the 2D implementation, as shown in Fig. 12(a). The systems share the same IO driver cells to communicate with the outside world. An IO multiplexing module is placed in between the two systems, which allows either one of the two systems to use the driver cells to communicate. The die-level power delivery network is designed on M7. As both systems have their PG rings on M5-M6, horizontal M7 stripes are used to connect these

rings with the IO ring around the die. Several stripes on M7 are used to ensure reliable power delivery to both systems.

2.6.3 Sign-off Verifications

DRC verifications are performed on chiplet designs and the assembled system designs. After fixing system-level design errors, both systems are merged together into the die-level GDS. Fig. 12(b) shows the combined GDS. To pass the sign-off verifications of the foundry, some adjustments had to be performed to the final design. E.g., to pass the antenna rule check, some of the wide wires on M8 (RDL1) and M9 (RDL2) were adjusted to reduce the antenna area. To fulfill the density requirement, special filler cells and metal fills were used. After all tests are successfully passed, the design was sent out for fabrication. Fig. 12(c) shows the microscopic die-shot of the fabricated die.

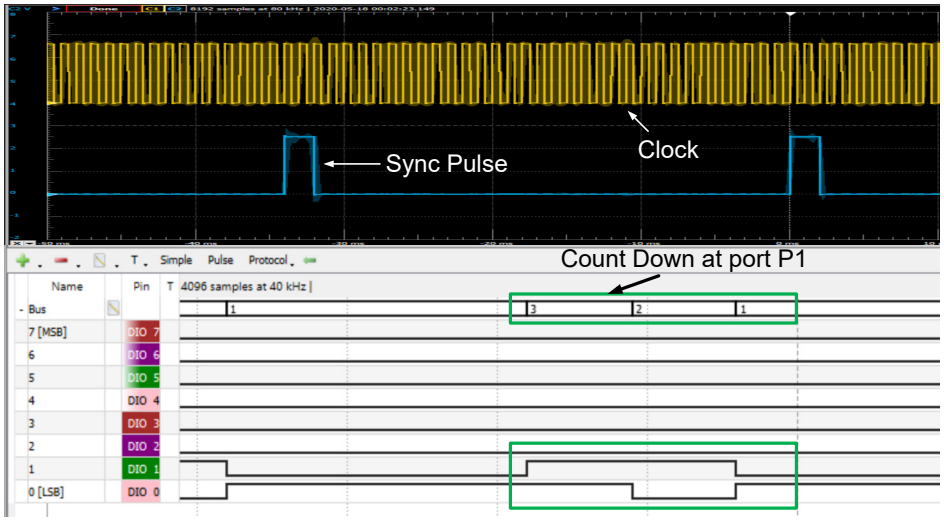


Figure 13: Chip testing waveforms from the logic analyzer

2.6.4 Chip Testing and Flow Validation

Both systems in the fabricated chip are tested using test vectors generated using a logic analyzer. Among several test cases, Fig. 13 shows the waveforms of the GPIO countdown test. In this test, the system reads the GPIO for a top value and then counts down to 1. At the end of the

countdown, it generates a pulse (Sync Pulse) at a specific pin. The clock signal, Sync Pulse, and the countdown values on a bus are shown in Fig. 13. Both systems in the die are tested individually, and both of them passed all the tests successfully. This silicon design proves that the holistic flow is fully compatible with the current ASIC CAD flow and foundry model. The agile design approach can make designing custom 2.5D multi-chiplet systems as easy as designing 2D modular ASICs.

Chapter 3

In-Context Methodologies

Although the holistic design method is powerful and adaptive to any technologies, one fundamental issue is that it cannot be applied to heterogeneous systems using standard ASIC tools. The heterogeneity consists of multiple chiplets that are implemented in different technologies. For example, AMD designed a processor Core-Chiplet in 7nm with an IO chiplet in 14/12nm technology.

Since the holistic flow presented in Chapter 2 requires to assemble the chiplets into a unified design environment, it cannot be applied to heterogeneous systems where the device stacks are different. At the present, no standard tool flows support heterogeneous technology files into a single physical design environment. Therefore, in-context design method is developed, where physical design steps involving heterogeneous technologies are performed separately, yet maintaining a holistic view in design, analysis, and optimization steps. This is achieved through creation of design context for a small part of the 2.5D system, and performing analysis on it. Later, all such contexts are carefully combined together to create a holistic view for system-level analysis and optimization purposes.

In this chapter, three flavors of the in-context chiplet-package co-optimization flows are presented. These flavors of the in-context flow are,

1. A scalable per-chiplet in-context flow,
2. A highly accurate per-technology in-context flow, and
3. A timing-accurate scalable in-context flow,

Like the holistic flow, these flows incorporates the features required to achieve the design goals in a high-density 2.5D integration technology. The in-context flows presented here are completely compatible with all standard ASIC tools for design, extraction, and analysis.

Table 6: Parameters (in μm) of the modified Nangate45 PDK routing layers

	M6	via6	M7	via7	RDL1	viaR1	RDL2	viaR2	RDL3
Height	2.28	3.08	3.9	7.5	12.5	17.5	22.5	27.5	32.5
Thickness	0.8	0.82	3.6	5	5	5	5	5	5
Width	0.4	0.4	2	5	10	10	10	10	10
Spacing	0.4	0.44	2	10	10	20	10	20	10

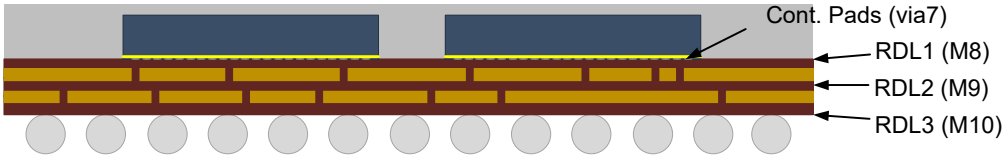


Figure 14: Package redistribution layer stack of the modified Nangate45 PDK

3.1 Design and Technology Settings

For illustration purposes as well as case study, the micro-controller system presented in Chapter 2 is designed in Nangate45nm PDK. In the implementation of Chapter 2, the 16KB memory system has four 4KB banks. For the study of in-context flows, the 4KB banks are further subdivided into four 1KB memory blocks. With such a granular design of the memory system provides more options while performing partition and floorplan. The system architecture and chiplet partitions of the micro-controller system are shown in Fig. 15. The Core-Chiplet contains all the logic blocks and 8KB memory while the Mem-Chiplet contains only the rest 8KB of the memory. OpenRAM [29] memory compiler is used to compile the 1KB memory module with a one-byte word size.

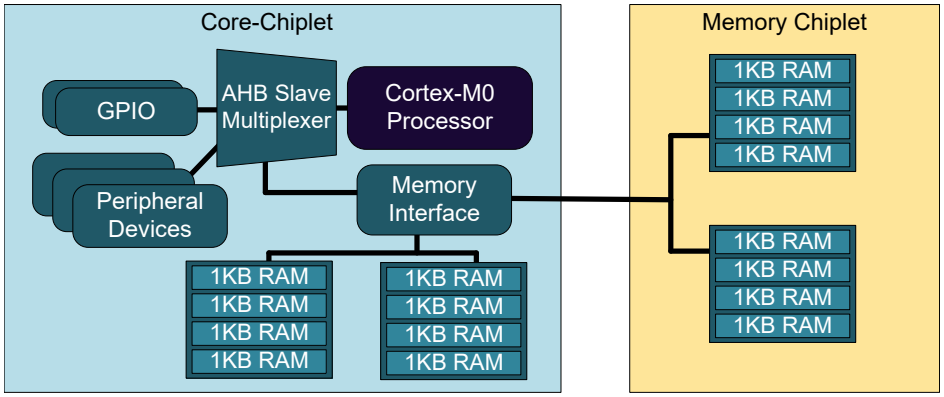


Figure 15: System architecture and chiplet partitions of the Cortex-M0-based reference design

For comparative study, the designs implemented in in-context flows are compared with a version implemented in the holistic flow. Currently, standard chip design tools do not support package routing layers. To perform holistic planning and verification of a 2.5D system, the chiplet and package designs need to be loaded in the same design environment. For this reason, the Nangate45nm technology is modified to support both chiplets and the package designs together in a chip design environment. The PDK is modified to create two technology stacks, named 7M3R and 6M3R. In 7M3R, the layers M1-M7 are used for chiplet internal routing. The top three layers, M8-M10, are adjusted to mimic TSMC 2.5D InFO package routing layers. Table 6 and Fig. 14 together describe our settings for the package layers. The 6M3R stack has six lower layers with the same dimensions as the corresponding layers of 7M3R for chiplet internal routing. The three RDLs are exactly the same as in 7M3R. Although both of these stacks are for 45nm technology from device perspective, they are heterogeneous from the tool flow perspective.

3.2 Holistic Reference Designs

The designs in Chapter 2 are implemented in TSMC 65nm PDK. As a result, the in-context designs implemented in Nangate45nm PDK in this Chapter cannot be directly compared with them. For the sake of comparative study, all the design cases presented in Chapter 2 are re-implemented here using the 7M3R technology. Table 7 summarizes the analysis results of the design cases. The experimental designs implemented using the in-context flows are directly compared with these design cases in the later sections of this Chapter.

3.3 Per-Chiplet In-Context Flow

This is the first version of the in-context flow. In this flavor, a package context is created for every chiplet separately. The design, analysis, and optimization of the chiplet is performed taking into account the package context. After each iteration of physical design, parasitic extraction is performed on a sub-design containing the chiplet and its package context. Finally, these contexts are stitched together for system-level analysis and optimizations.

Table 7: Analysis results of reference designs implemented in the holistic flow

Design Case	Case-1	Case-2		Case-3 first iteration		Case-3 2nd/final iteration	
Chip Design	2D Chip	Core Chiplet	Mem. Chiplet	Core Chiplet	Mem. Chiplet	Core Chiplet	Mem. Chiplet
Logic Gates#	17595	17783	132	17915	148	18214	45
Buffer/Inverter#	3700	2740	132	2865	148	2955	45
Die Size (µm×µm)	550×550	390×590	350×470	390×590	350×470	390×590	350×470
Total Chip Wirelength (mm)	412.9	350.9	40.14	361.2	45.07	366.3	41.99
M6 Wirelength (mm)	79.94	30.81	5.986	31.86	8.201	31.42	8.445
M7 Wirelength (mm)	0	1.783	0.598	1.875	0.589	2.02	0.624
Max Frequency (MHz)	333	245		280		300	
Performance Gap	0%	100%		60.23%		37.50%	
Chip Power (mW)	10.6	7.751	0.194	9.043	0.216	9.840	0.162

3.3.1 Chiplet-Package Co-Design Flow

The flow is illustrated in Fig. 16. The first step is to create in-context designs as another level of design hierarchy. The context of a chiplet should include the area covering the whole chiplet and necessary neighboring regions. This ensures all chiplet-package interactions are considered during the extraction. Note that each in-context chiplet can be implemented with different technology files. Therefore, heterogeneous systems are partitioned into several sub-designs, where each one is an extended 2D design.

Once all bare chiplets are converted into the in-context chiplets, a top-level design is generated to connect the individual in-context chiplets into a merged system. However, as the top-level design does not need details within each in-context chiplets, only RDL routing layers are included in the design. This hides the device layer to the top-level, thus entire heterogeneous systems can be assembled. Standard extraction tools can then be used to perform extraction on each in-context design and the top-level, and the SPEF files are stitched with hierarchical annotations.

This flow is implemented using a custom RDL planning tool. It partitions the floorplan and RDL routing into each chiplets and creates the corresponding design hierarchy with Verilog netlists. Then, the RDL wires are imported during the chiplet implementation to form the in-context chiplets. The extracted SPEF files are then merged using an STA tool. To validate the in-context extraction, the best strategy is compare it against the holistic method. Since the holistic method can only be applied to homogeneous systems, the in-context design extraction is performed on the homogeneous system which is designed using the holistic design method. Two in-context chiplets are created all in the modified Nangate45 with seven metal and three RDL layers (7M3R).

The extraction results are compared in Table 8. As results show, the in-context extracted ground (GCAP) and coupling (CCAP) capacitance are highly close to the holistic extraction on all chiplet layers, even though the chiplets only see a partial design of the package. The total ground capacitance between holistic (25550 fF) and in-context (25367 fF) extraction is only

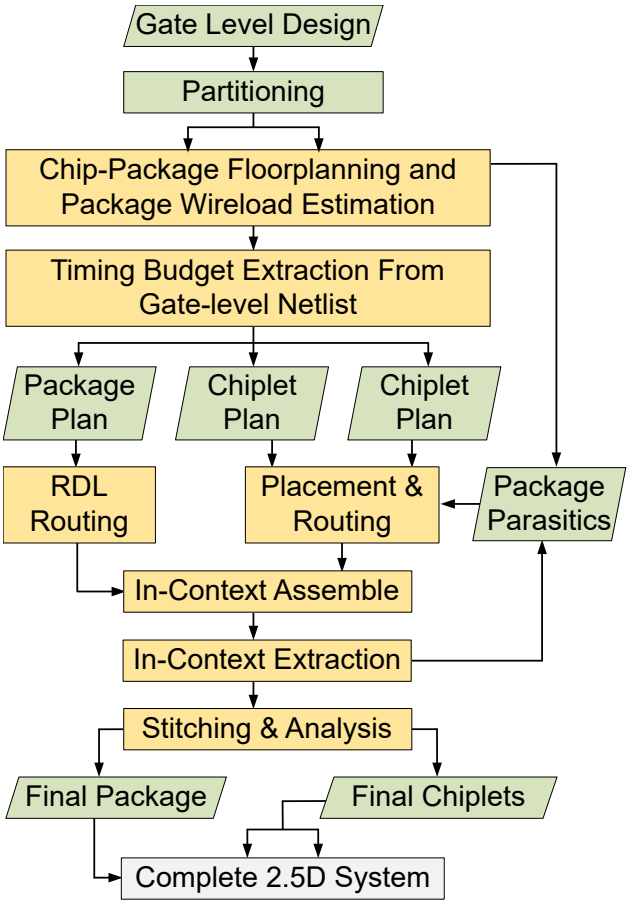


Figure 16: Per-chiplet in-context flow for heterogeneous systems

0.71% while coupling capacitance difference between holistic (16278 fF) and in-context (16406 fF) extraction is only 0.79%. However, the package layer capacitance are slightly overestimated, especially on the ground capacitance. This is mostly because of the RDL partition impact. With RDL wires separated into multiple designs, the fringe capacitance is computed on the cutting edge of the package wires. However, as these cutting faces do not exist in the holistic design, the capacitance are extracted correctly.

3.3.2 Experimental Study

Since the in-context design strategy targets heterogeneous systems, to demonstrate its capability, a Mem-Chiplets in a different PDK is designed. However, since the OpenRAM compiler only supports a limited selection of technologies, the same Mem-Chiplet is implemented using gscl45

Table 8: Comparison of Holistic (Holi) vs In-Context (In-C) ground (GCAP) and coupling (CCAP) capacitance extraction results (in fF) of Case-3 final homogeneous design.

Metal Layer	M1-M5	M6	M7	R1	R2	R3
In-C GCAP	21119	2053	273	1103	306	696
Holi GCAP	21119	2054	272	1040	247	636
In-C GCAP Err	0.00%	-0.01%	0.09%	6.03%	24.0%	9.46%
In-C CCAP	9171	1265	153	1563	2489	1765
Holi CCAP	9172	1263	156	1544	2421	1721
In-C CCAP Err	-0.01%	0.17%	-2.10%	1.20%	2.81%	2.56%

cell library which is bundled with the FreePDK45. Instead of using the same 7M3R homogeneous technology, the Mem-Chiplet only uses six metal layers (6M3R). The Core-Chiplet remains the same with the 7M3R Nangate45 PDK, forming a heterogeneous system with the Mem-Chiplets. Fig. 17 shows the layout of the heterogeneous systems with two in-context chiplets.

Table 9: In-Context heterogeneous design results with 7M3R Core Chiplet in Nangate45 and 6M3R Mem Chiplet in gscl45.

Design iteration	LPD (ns)	Max Frequency	
with RDL wireload	3.55	281 MHz	
In-Context 1st iteration	3.35	298 MHz	
In-Context 2nd/final	3.35	298 MHz	
Final Design	Wire	Cell	Total
In-Context Power (mW)	4.29	6.22	10.51

In-context extraction is performed on each chiplets and the timing optimization flow is followed similar to the holistic design flow. As observed from Table 9, in-context extraction has successfully enabled the iterative timing optimization on the heterogeneous designs. The designs with RDL wire-load estimation and final iteration of Table 9 correspond to the Case-3 first and final iterations of Table 7, respectively. Comparing these designs, we can see that in-context designs achieve the same performance as the holistic designs in the corresponding iterations. The minor changes in performance are caused by the small errors in the in-context extraction results as previously discussed. However, the results still prove that the in-context flow, which can support heterogeneous techonologies, is as effective as the holistic approach.

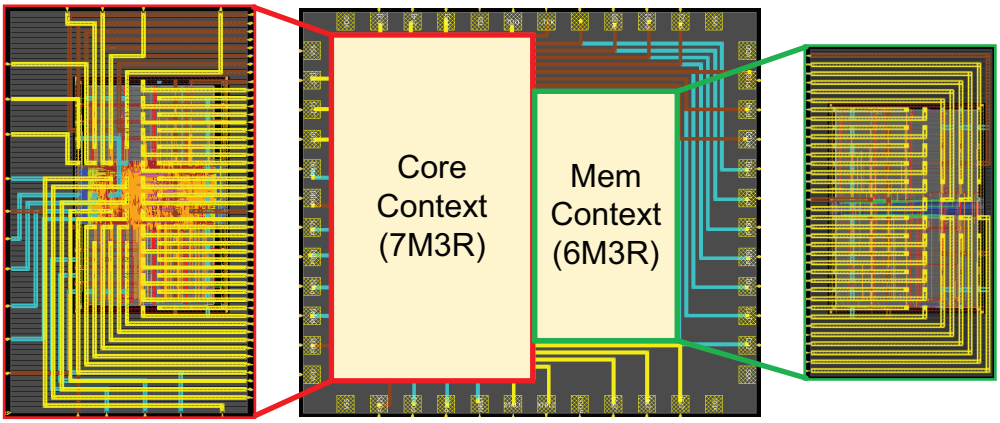


Figure 17: Layouts of the in-context chiplets for heterogeneous integration

3.4 Per-Technology In-Context Flow

The in-context extraction strategy presented in Section 3.3 is not accurate enough to ensure reliable analysis and timing context accuracy. The flavor of in-context flow presented in this section can achieve holistic-like accuracy and generate highly reliable analysis results and timing contexts for iterative optimizations. This flow offers a revised extraction strategy to perform in-context extraction of heterogeneous 2.5D systems through a new post-processing method to improve the accuracy of extraction and analysis results. A comparative study is presented to validate the effectiveness of this methodology.

3.4.1 Chiplet-Package Co-Design Flow

This flavor of the in-context flow is demonstrated in Fig. 18. Though the figure illustrates the flow for two heterogeneous technologies, the same methodology can be applied to a 2.5D system involving more than two technologies.

In the planning step, a holistic plan of the system is prepared. The gate-level netlist is partitioned into chiplets based on the system requirements. The initial package floorplan, inter-chiplet routing, and package wireload estimation is performed in this step. An RDL planning tool is developed, which generates this top-level plan. Using this plan, a hierarchical design is prepared, where the package is treated as the top-level design with chiplets as

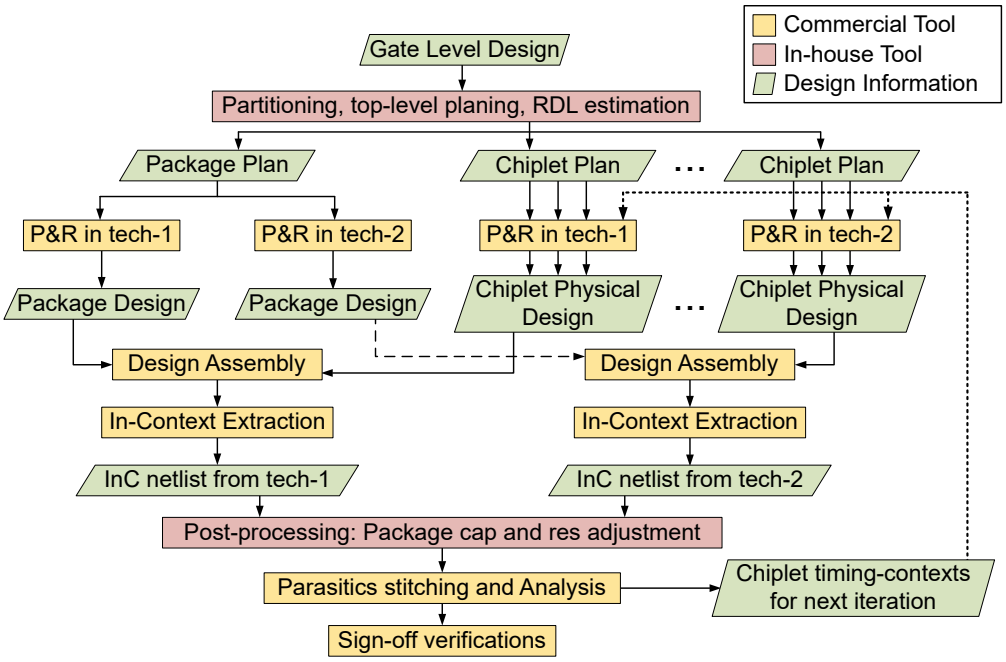


Figure 18: Per-technology in-context co-optimization flow for heterogeneous 2.5D systems

sub-designs. After this hierarchical sub-design formation, chiplets and the package can be implemented independently in parallel.

The physical design of chiplets is carried out as if they were individual 2D chips, with some additional constraints due to the top-level plan. Shown in Fig. 18, each chiplet has its own separate plan and can be implemented in different technologies and configurations, which is highlighted using the parallel arrows in the figure. The package design is implemented once for each of the heterogeneous technologies. A unified technology stack is generated combining the routing stack of the chiplet technology and package RDLs. The package design is implemented as per the top-level plan. The chiplet routing is generated using the RDL planning tool employing a greedy strategy.

3.4.2 In-Context Extraction and Post-Processing

After the physical design of chiplets, DRC is performed on them. Then, they are assembled with the package design for extraction. In-context extraction is performed per technology to capture the cross-boundary interactions among chiplets and the package. As presented in Fig. 18, chiplets

from the same technology are assembled with the package for extraction, while chiplets from other technologies are treated as blackbox macros. After this step, there is one parasitics netlist per technology.

The in-context extraction results cannot be directly used to create a holistic view of the parasitics. This is because the entire package is included in each of all the netlists. The parasitic netlists need to be adjusted for double-counting package wires. In the experimental study, the top-level package parasitics is extracted treating all chiplets as blackbox macros. This study reveals that for two technologies combined layer-wise, the overestimation of ground and coupling capacitances on the package nets are exactly equal to the top-level package parasitics. Based on this finding, a tool was developed that can adjust the in-context parasitic netlist based on the top-level package parasitics.

This tool reads an in-context chiplet parasitics, the top-level package parasitics, and a user-defined factor to perform adjustments on each net and computes the in-context parasitics using (1) and (2). These equations reduce the ground and coupling capacitances of the package nets in the in-context parasitics by a user-defined fraction (*userFact*) of the top-level package capacitance. As with incremental parasitic annotation, some resistive nodes are annotated twice, the resistance values of the package wires are also doubled in each in-context netlist. As a result, in the annotated parasitics, the parallel equivalent resistance remains unchanged. Though these adjustments are performed per-node based on the total capacitance per-layer, this method generates highly accurate parasitics per-net. This claim is validated through the experimental study in Section 3.4.3.

$$layerFact_x = \frac{CapRDL_x - userFact \times TCapRDL_x}{CapRDL_x} \tag{1}$$

$$newNodeCap = nodeCap \times layerFact_x \tag{2}$$

Here,

Parameter	Definition
$CapRDL_x$	Total ground (coupling) capacitance on package layer number (pair) x of the net in the in-context parasitics
$TCapRDL_x$	Total ground (coupling) capacitance on package layer number (pair) x of the net in the top-level package parasitics
$userFact$	User specified factor ($0 < userFact \leq 1$)
$layerFact_x$	Calculated adjustment factor for all ground (coupling) nodes of the net on layer number (pair) x
$newNodeCap$	The value of the capacitance node in the adjusted in-context parasitics netlist

These adjusted in-context parasitics are incrementally annotated in the timing analysis tool to create a holistic view of the complete system parasitics. After analysis, timing contexts for all chiplets are exported from the analysis tool. These contexts have a detailed view of the entire system and can be used for cross-boundary optimizations with a tighter timing budget to improve the system performance. These contexts are used to re-implement the chiplets. This iterative approach is highlighted using the dotted line on the right-half of Fig. 18. Several iterations of chiplet physical design, assembly, extraction can be performed to optimize the system performance.

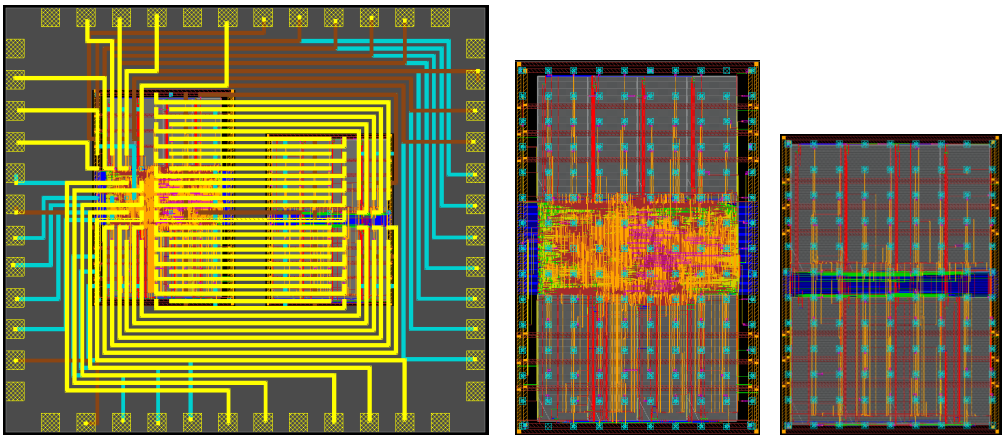
Table 10: Coupling and ground capacitances (in fF) between routing layers in holistic extraction

	M1-M5	M6	M7	RDL1	RDL2	RDL3
M1-M5	5990	473.1	39.19	57.84	10.19	6.732
M6	473.1	582.2	89.56	124.4	12.27	9.677
M7	39.19	89.56	51.21	17.84	1.789	2.574
RDL1	57.84	124.4	17.84	301.1	1012	38.43
RDL2	10.19	12.27	1.789	1012	296.7	1078
RDL3	6.732	9.677	2.574	38.43	1078	512.2

Ground Capacitance						
Metal Layer	M1-M5	M6	M7	RDL1	RDL2	RDL3
Capacitance	21605	2161	284	1032	219	513

3.4.3 Experimental Study

For comparative study, the homogeneous system is implemented using both holistic flow and this in-context flow. In the first iteration of the chiplet physical design, the chiplets are implemented using top-level constraints and estimated package wireload. Fig. 19(b)–(c) shows these chiplets finished designs. In the holistic design, both chiplets are assembled with the package, as shown in Fig. 19(a), and holistic extraction is performed. This holistic extraction method is used to perform iterative optimization of the chiplets. From here on, this design is referred as "Homogen-Holi" design. In the in-context design, only one chiplet is assembled at a time, and in-context extraction is performed on the assembled design. These in-context parasitics are adjusted using the methodology discussed in Section 3.4. These in-context parasitics are used in the iterative optimization of the system. From here on, this design is referred as "Homogen-InC" design.



(a) Assembled System for Extraction (b) Core-Chiplet (c) Mem-Chiplet

Figure 19: Chiplets and assembled package layouts of the homogeneous 2.5D system

Until the first extraction, both Homogen-Holi and Homogen-InC designs are basically the same design. So, their extraction results can be compared to verify the accuracy of the in-context extraction flow. Table 10 shows the holistic extraction result performed on Homogen-Holi design after the first implementation of the chiplets. As observed from the table, there exists a significant coupling between the routing layers at the chiplet-package boundary. The detailed interactions between chiplets and the package are captured in the extraction result. For example, though M7 is

the top-most chiplet routing layer, the coupling between RDL1 and M6 is greater than that with M7. This is because, in the chiplet designs, there are significantly fewer wires on M7 compared to that on M6.

Table 11 presents the comparison between the extraction result of the holistic flow, the per-technology in-context extraction methodology, and the per-chiplet flavor of the in-context flow. The extraction error in the per-chiplet flavor varies between -2.10% to 24.0%. This much error cannot ensure reliable analysis results. In this flavor, the extraction error in both ground and coupling capacitance is less than 1%, which is a significant improvement over the per-chiplet flow. In the parasitics adjustment tool, 0.4 with the Core-Chiplet context and 0.6 with the Mem-Chiplet context are used for the value of *userFact* in (1) for this design. The initial setting was to use 0.5 for both contexts. But, as the chiplets are of different sizes, the overestimation due to the package is not equal on both chiplets. This methodology can be explored further to calculate this factor from design information.

Table 12 shows the per-net accuracy of timing analysis and context using the adjusted in-context parasitics. The error is calculated w.r.t the holistic analysis result. As observed from the table, this extraction result can achieve 99.4% accuracy in both timing analysis and timing context generation for iterative optimizations. Table 13 shows the power and performance of different iterations of Homogen-Holi and Homogen-InC designs. As observed, the two designs match closely in all iterations. These results validate that the in-context flow is accurate and effective in designing high-performance heterogeneous 2.5D systems with very high reliability.

3.4.4 Heterogeneous Design Case-Study

Here, a design case-study of a heterogeneous system using 45nm technology is presented. This is the same two-chiplet microcontroller system. However, in this design, the Mem-Chiplet is implemented using 6M3R and standard cells from the gscl45 cell library, which is bundled with the FreePDK45. Core-Chiplet is implemented in 7M3R the same as before. After top-level planning, the first implementation of the chiplets is performed using timing contexts generated

Table 11: Comparison of Holistic (Holi) vs. In-Context (In-C) ground (GCAP) and coupling (CCAP) capacitance extraction (in fF)

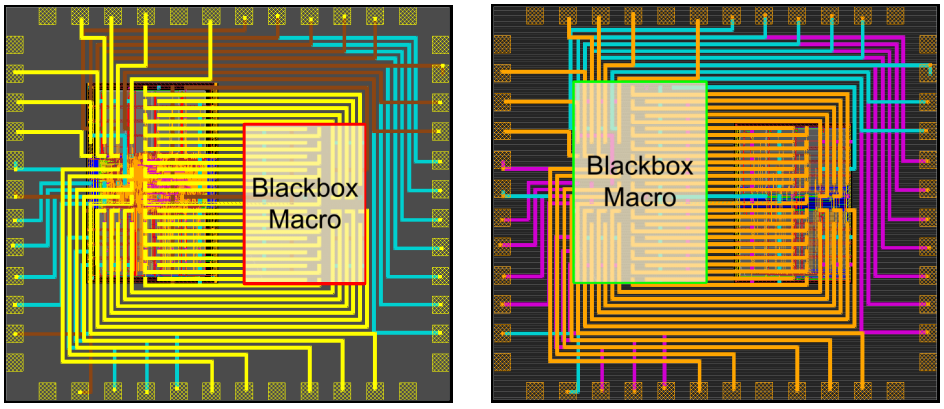
Metal Layer	M1-M5	M6	M7	R1	R2	R3
In-C GCAP	21605	2162	284	1034	220	513
Holi GCAP	21605	2161	284	1032	219	513
In-C GCAP Err (per-tech)	**0.00%**	**0.00%**	**0.01%**	**0.24%**	**0.6%**	**0.00%**
In-C GCAP Err (per-chip)	0.00%	-0.01%	0.09%	6.03%	24.0%	9.46%
In-C CCAP	8988	1292	203	1553	2412	1648
Holi CCAP	8989	1291	202	1553	2412	1648
In-C CCAP Err (per-tech)	**0.00%**	**0.04%**	**0.64%**	**0.03%**	**-0.01%**	**0.00%**
In-C CCAP Err (per-chip)	0.01%	0.17%	-2.10%	1.20%	2.81%	2.56%

Table 12: Per-net accuracy comparison of inter-chiplet package wires

Parameter	Max. Error	Min. Error	Avg. Error
Path delay	3.30%	0.00%	0.61%
Design constraint	1.80%	0.30%	0.62%
Load Capacitance	1.70%	0.00%	0.29%

through timing analysis on the gate-level netlist. The estimated package wireload is appended with this timing context. The top-level package design is implemented in both 7M3R and 6M3R stacks. After DRC, chiplet physical designs are assembled with their corresponding top-level package design, keeping other chiplet as blackboxes. Fig. 20 shows the assembled design contexts of both chiplets. In-context extraction is performed on each assembled design. For each stack, the extraction on the top-level package is also performed. In the post-processing step, in-context parasitics are adjusted using the top-level package parasitics from the corresponding technology stack. In this design, the same *userFact* values are used in (1) as in the Homogen-InC design of Section 2.4. These in-context parasitics are used for timing analysis and context generation. After two such iterations with in-context extraction, no further improvement is observed in the system performance.

Table 13 compares the performance of this design with the homogenous designs of Section 3.4.3. This design is referred to as “Heterogen-InC” in the table. The performance results of all design iterations are very close to that of the holistic design. This is because all these designs are using the same device node even though they are implemented in different routing



(a) Assembled Core-Context (7M3R) (b) Assembled Mem-Context (6M3R)

Figure 20: Layouts of the assembled heterogeneous system for in-context extraction

technologies. The second part of the table compares the power figures of the final iteration of the designs. There is some difference in the power figures in the Heterogen-InC design because the Mem-Chiplet uses different cells and numbers of routing layers. Despite the differences, the power figures are comparable with the homogeneous designs. These results prove this flow can optimize heterogeneous 2.5D systems to achieve holistic-homogeneous system-like performance if that is feasible.

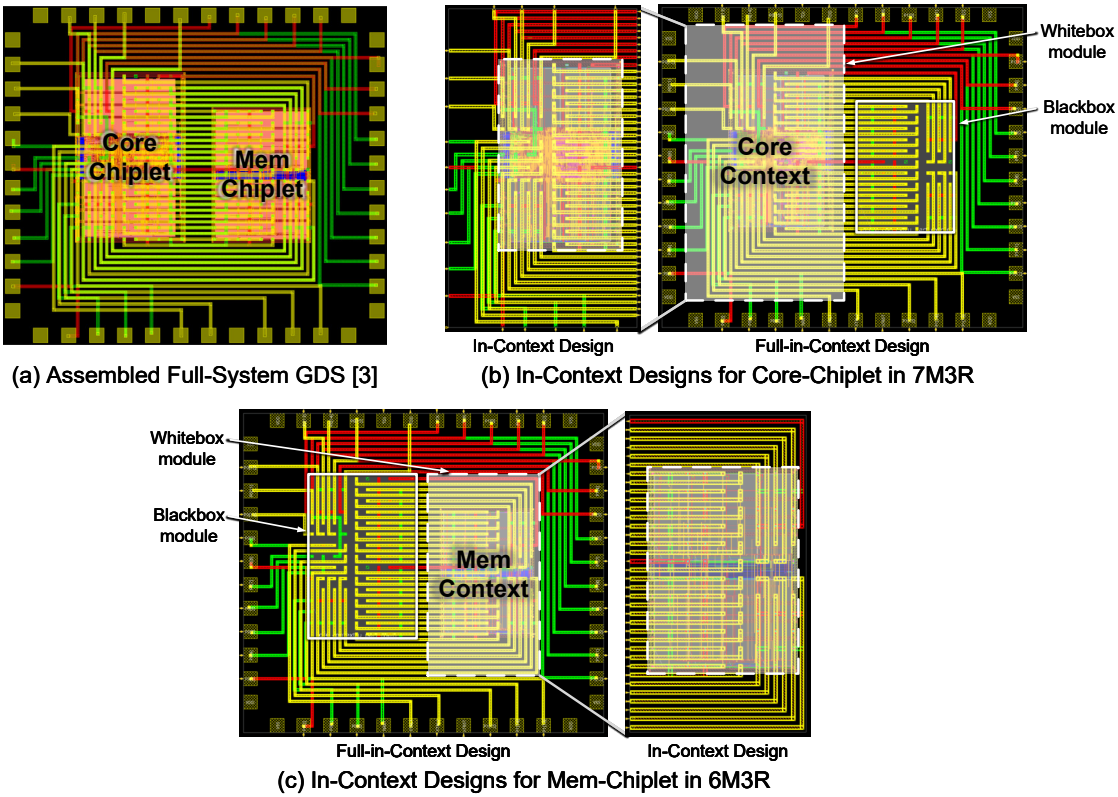
Table 13: In-Context heterogeneous design results with 7M3R Core-Chiplet in Nangate45 and 6M3R Mem-Chiplet in gscl45.

Performance Comparison (MHz)			
Design iteration	Homogen-Holi	Homogen-InC	Heterogen-InC
With RDL wireload	288	288	287
In-Context 1st iteration	293	294	294
In-Context 2nd/final	300	300	300
Power Comparison of the Final Iteration (mW)			
Power Group	Homogen-Holi	Homogen-InC	Heterogen-InC
Wire	4.34	4.30	4.24
Cell	6.35	6.37	6.22
Total	10.69	10.67	10.46

3.5 Timing-Accurate Scalable In-Context Flow

In this section, a scalable in-context chiplet-package co-optimization flow for heterogeneous 2.5D systems is presented. This flow leverages industry-standard tools to perform in-context extraction

on heterogeneous 2.5D systems while ensuring the compatibility of extraction results with the industry-standard ASIC CAD flow. The holistic-like extraction result is utilized to perform cross-boundary analysis and iterative system-level optimization. In flow offers an accurate and scalable extraction strategy to perform in-context extraction perform timing and signal integrity analyses with a complete view of the heterogeneous system. A comparative case study is presented to validate this methodology.



(a) Assembled Full-System GDS [3]

(b) In-Context Designs for Core-Chiplet in 7M3R

(c) In-Context Designs for Mem-Chiplet in 6M3R

Figure 21: Package and assembled system layouts of the experimental homogeneous and heterogeneous 2.5D systems.

3.5.1 Chiplet-Package Co-Design Flow

This flavor of the in-context flow is illustrated in Fig. 22. The top-level planning consists of architecture partitioning and RDL planning steps in the figure. Based on system requirements, the gate-level netlist is partitioned into chiplets. These chiplets are converted into sub-designs, treating the top-level design as the package design. As there is no physical design to extract the

parasitics in this step, the package wireload is estimated using a model by the RDL planner tool. Based on this plan and estimated wireload, timing budgets are extracted for each chiplet. In-context partitions are defined for each chiplet, which contains the chiplet and part of the package surrounding it. Fig. 21 (b), (c) illustrate such in-context partitions. Hierarchical sub-designs are generated for in-context partitions and chiplets with top-level constraints.

The top-level planning step determines the package floorplan, inter-chiplet routing, and signal assignments of chiplets. This plan is followed in the physical implementation of the package. The physical designs of chiplets are prepared by treating them as individual chips with top-level constraints. The top-level constraints ensure that individual chiplet designs conform to the holistic plan and design budgets. Each chiplet has its own separate plan and can be implemented in any technology, independent of other chiplets.

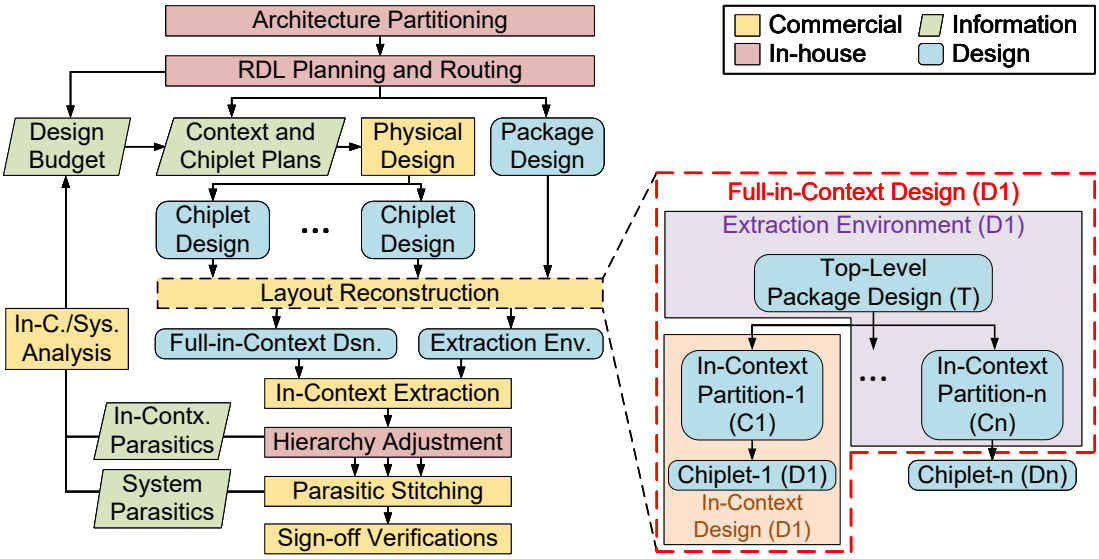


Figure 22: Timing-accurate scalable in-context flow for heterogeneous 2.5D systems.

3.5.2 In-Context Parasitic Extraction

This flow takes advantage of the industry-standard in-context extraction tool designed for flip-chip package extraction. Flip-chip extraction tools take the chip design as the extraction target and the package routing as the extraction environment. The coupling capacitance between

chiplet and package wires is converted to ground capacitance. Although this extraction result is good enough for chip-level timing analysis and optimizations, the chiplet-package interactions are lost. As a result, system-level analysis and optimizations like STA, signal integrity (SI), and power integrity (PI) analysis are not possible.

Fig. 22 (b) shows the design hierarchy in this flow. In the layout reconstruction step, different levels of the hierarchy are assembled to create layouts for extraction. The assembly of a chiplet, D1, with its in-context partition of the package is referred to as the "in-context design" of D1. The assembly of all in-context partitions, except that of D1, with the top-level package is used as the "extraction environment" of D1. Note that the extraction environment does not include chiplet details and treats them as black boxes. A combination of the in-context design and its extraction environment is the "full-in-context design" for the chiplet. Fig. 21 (b), (c) show these layouts for our experimental design. For top-level package T, any chiplet D_i, its in-context partition C_i, and a given chiplet D_x, general mathematical definitions of these designs are given below, where summation represents design-assembly.

$$\begin{aligned} \text{In-Context Design:} &\quad C_x + D_x \\ \text{Extraction Environment:} &\quad \textstyle\sum_{i=1, i\neq x}^{n} C_i + T \\ \text{Full-in-Context Design:} &\quad \textstyle\sum_{i=1}^{n} C_i + T + D_x \end{aligned}$$

In this flow, the full-in-context design of a chiplet and its extraction environment are used with the flip-chip extraction tool. The tool performs extraction on the entire in-context design instead of the chiplet only. As a result, the chiplet-package interactions within the in-context design are preserved in the parasitic netlist.

Since flip-chip extraction tools are not designed for hierarchical extraction, the extracted parasitic netlists cannot be directly used for hierarchical annotation. A custom tool is developed to fix this hierarchy problem, which adjusts the terminal nodes of the inter-chiplet package nets based on the design hierarchy. It also performs some clean-up to remove additional information related to other chiplets that are not part of the extracted in-context design.

The extraction flow creates separate parasitic netlists for each in-context design. As each

netlist contains all chiplet-package interactions for a given chiplet, it can be used to perform cross-boundary analysis and optimization at the in-context design level. Moreover, the netlists can be stitched together to create a holistic view of the system to perform system-level analysis. In the experimental study, the system-level parasitic netlist is used to perform full-system STA and create timing contexts for each chiplet. These timing contexts are used to perform iterative optimization of the chiplets to improve overall system performance. Similar iterative optimizations can be performed to improve the package design, SI, and PI of the entire system.

3.5.3 Experimental Study

For a comparative study, a homogeneous system using the holistic flow as well as this in-context flow is implemented. This system is designed using the 7M3R technology and standard cells from the Nangate45nm cell library. Both chiplets are assembled with the package for holistic extraction. In the in-context flow, in-context partitions of the package are created and the extraction methodology discussed in Section 3.5.2 is followed. Fig. 21 (a) shows the assembled GDS of the homogeneous system. To study the compatibility and effectiveness of this flow with heterogeneous systems, the microcontroller system is implemented using two different PDKs. The Core-Chiplet is implemented in 7M3R using cells from Nangate45nm cell library, and the Mem-Chiplet is implemented in 6M3R using cells from the gscl45 cell library, making the design heterogeneous.

Table 14 presents the comparison between the extraction result obtained using the holistic methodology, this in-context methodology, and the per-chiplet in-context methodology presented in Section 3.3. In this section, this flavor of in-context flow is referred as the “new flow” and the flow in Section 3.3 as the “old flow.” As observed from the table, the coupling capacitance between chiplets and the package is preserved with almost holistic-like accuracy, and is comparable to the numbers of the old flow. The coupling numbers for chiplet routing layers (M1-M7) are within +/-3%. This accuracy level is good enough to perform chiplet-level SI analysis, including the impact of RDLs.

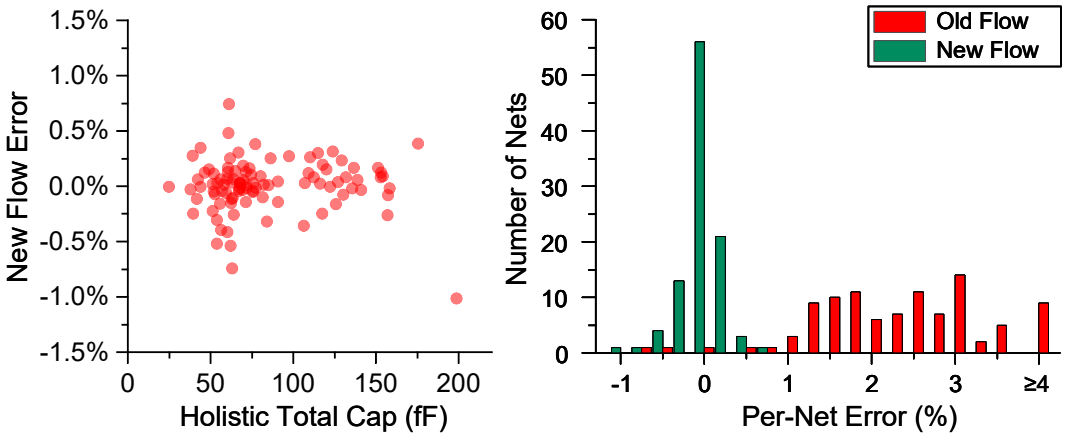


Figure 23: Comparison of the total capacitance on individual nets of the this flavor of the flow with per-chiplet in-context flow

Table 14: Comparison of holistic (Holi) vs. in-context (In-C) coupling (CCAP) and total (Total CAP) capacitance extraction (in fF)

	Metal	M1-M5	M6	M7	R1	R2	R3
CCAP	Holi	9275	1172	196	1529	2441	1685
	In-C Old	9346	1181	188	1564	2478	1690
	In-C New	8992	1203	193	1517	2390	1640
Total CAP	Holi	31056	3307	498	2547	2669	2209
	In-C Old	31140	3324	489	2661	2749	2251
	Old Err%	**0.27%**	**0.51%**	**-1.79%**	**4.49%**	**3.01%**	**1.91%**
	InC New	31238	3350	495	2591	2654	2192
	New Err%	**0.59%**	**1.31%**	**-0.59%**	**1.74%**	**-0.55%**	**-0.76%**

Unlike the old flow, the total capacitances on all layers in the new flow are very close to that of the holistic extraction. In the old flow, the total capacitance is highly overestimated on the package layers, due to additional fringe capacitances extracted at the boundary of the in-context partitions. These fringe capacitances are non-existent in the actual design. In this new flow, because the extraction tool is aware of the extraction environment while performing in-context extraction, those fringe capacitances are not extracted at the boundary.

In Fig. 23, the scatter plot on the left shows the total capacitance error of each net in the new flow with respect to the holistic extraction. As observed in the scatter plot, the extracted parasitics on each net is as accurate as the holistic flow, with 1% error margin. The histogram on the right organizes the errors into 0.5% bins. The per-net extraction error in the new flow is 0% for most of

the nets. However, the parasitics is overestimated in almost all nets in the old flow, with error varying between -1% and 7%. Thus, this new flow improves the per-net extraction accuracy from 93% to 99% compared to the old flow. As the signal delay depends on the total load capacitance, the parasitic netlist obtained in this flow can be used to perform a highly accurate timing analysis of the system.

The iterative timing optimization results are shown in Table 15. As observed from the "Homogeneous" column, the timing optimization results from the in-context flow very closely match holistic flow. As the heterogeneous design incorporates different PDKs and cell libraries, the first implementation with RDL wireload would not match with the holistic designs. However, the optimization results of the following iterations closely match. Similar results are observed in the power comparison table. Both homogeneous designs have almost the same power numbers in the final iteration. As the heterogeneous design uses a different cell library in the Mem-Chiplet, the power numbers slightly differ from that of the homogeneous design. These results validate that this in-context flow for heterogeneous systems achieves the accuracy and optimization results comparable to the holistic flow for homogeneous designs.

Table 15: Comparison of holistic and in-context flow optimization results of the experimental designs

	Design	Homogenous		Heterogeneous
Performance	Iteration	Holistic	In-Context New Flow	
	Initial	288 MHz	287 MHz	278 MHz
	1st iteration	293 MHz	290 MHz	294 MHz
	2nd/final iteration	300 MHz	300 MHz	300 MHz
Power	Power Group	Holistic	In-Context New Flow	
	Wire	4.35 mW	4.37 mW	4.21 mW
	Cell	6.39 mW	6.36 mW	6.20 mW
	Total	10.74 mW	10.73 mW	10.41 mW

However, unlike the holistic and the per-technology flows, this flow is highly scalable in terms of the number of technologies and chiplets. As the in-context parasitic netlist contains chiplet-package interactions within the in-context partition, cross-boundary analysis and optimizations can be performed on each chiplet independently. In the end, a system-level holistic

view can be created through hierarchical annotation of the in-context parasitics to perform full-system analysis and verification. For a 2.5D system with multiple chiplet technologies, the initial plans can be distributed to several design houses with the package in-context partitions. They can perform cross-boundary analysis and optimizations in their own contexts without worrying about others parts of the package. This way, multiple design houses can collaborate on a large-scale heterogeneous 2.5D system, containing hundreds of chiplets in tens of heterogeneous technologies, yet maintain cross-boundary analysis, system-level optimization, and verification.

Chapter 4

Inductance-Aware Timing Optimization

The dimensions of interconnects on RDL have already reached the sub-micrometer range [16]. Due to chipletization and the use of Known-Good-Dies (KGD), it is now feasible to design a very large 2.5D system containing several chiplets. In large systems containing tens of chiplets, long RDL wires are unavoidable. These long RDL wires are going to exhibit significant inductive behavior. Traditionally, chip-scale interconnects are modeled using resistive and capacitive (RC) elements. Several previous studies have discussed the impact of inductive (L) elements of interconnects at the high-frequency range. It is observable that the signal oscillations in the RDL wires are significantly underestimated using only RC elements [30]. The study [31] also shows that properties of the driver also affect the oscillatory behavior of the voltage waveform. It is essential to take into account the inductive behaviors of the RDL wires to ensure system reliability and signal integrity of a high-performance 2.5D system with long interconnects.

Due to the complexity of chip routing, many inductance extraction methods proposed for package design are not able to handle dense and fine chip wires [32]. In this chapter, a chiplet-package co-optimization flow for 2.5D systems is presented, which takes into account the inductive effects of RDL wires on the system performance. This flow can automatically co-optimize the IO drivers and receivers between chiplets taking into account the timing overhead introduced by the inductive behavior of the RDL wires. Through SPICE simulation, an RLC interconnect delay model is developed to estimate the path delays through RDL interconnects. Custom tools are developed that work hand-in-hand with the existing commercial ASIC CAD tools to incorporate the inductance timing overhead in design, analysis, and optimization steps.

4.1 RLC Delay Modeling

As discussed in the previous studies [30, 31, 33], at high frequencies, the inductive effects of interconnects become significant on timing. Before a system can be optimized for the inductance

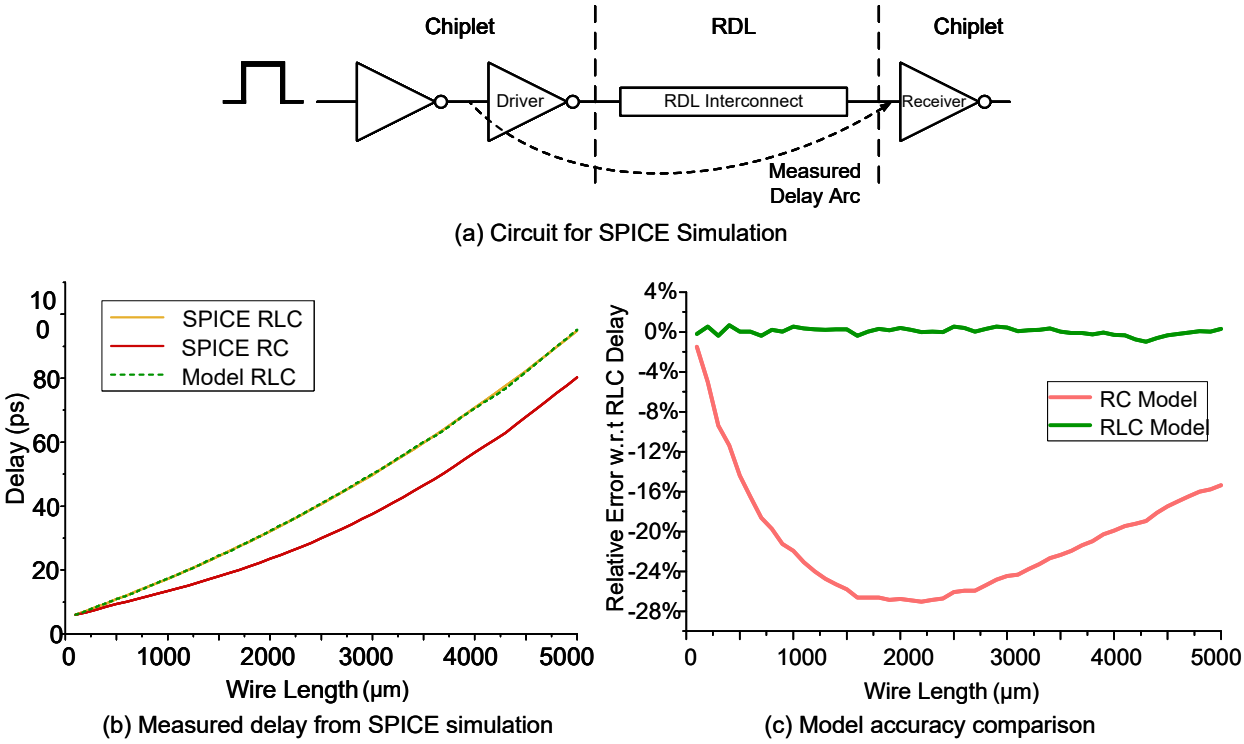


Figure 24: SPICE simulation and validation of the proposed model

overhead, the RLC interconnect delay need to be modeled accurately. The study presented in [33] modeled the global interconnect delay of 0.25 μm CMOS technology taking into account the inductive effects. Though the properties of 2.5D RDL and 0.25 μm CMOS global wires are different, the delay modeling methodology is utilized to develop an RLC delay model for the RDL wire drivers. Nangate45nm PDK is used as the standard cell library and RDL wire driver. The RLC delay model is developed based on this technology.

4.1.1 Interconnect Delay Study using SPICE

Compared to the chiplet internal wires, the RDL wires are wider and so have smaller resistance and larger capacitance per unit length. In the experimental design, RDL wires with width/spacing of 10 μm/10 μm are used. For the purpose of comparative study through SPICE simulation, the RDL wires are modeled with resistance 0.05 Ω/μm and capacitance 0.068 fF/μm. The inductance is modeled using the partial inductance of the wire. The partial inductance is calculated using the

following equation [34]:

$$L_{l,k} = \frac{\mu_0 l}{2\pi}\left[\ln(\sqrt{k^2+1}+k) - \sqrt{k^{-2}+1} + \frac{0.9054}{k} + 0.25\right] \quad (3)$$

For the partial self inductance, $k = l/r$, where l is the length of the RDL wire and r is the thickness of the wire. Referring to the TSMC InFO UHD technology [16], a thickness of 1 µm is used for the RDL wire. At 2 GHz, the skin depth of copper is approximately 1.45 µm. Thus, it can be safely ignored in this simulation. Fig. 24 shows the circuit used to perform SPICE simulation. A pulse source with 2 GHz frequency and rise and fall times of 10 ps is applied to the gate of an inverter, which drives the gate of the RDL wire driver cell. This driver cell drives a receiver gate connected to the other end of the RDL wire. In a real design, the drivers and receivers are supposed to be within chiplets and connected to the RDL wires through IO pads. These IO pads are ignored in this simulation. In multiple runs of the simulation, the RDL interconnect length is changed, and the simulation is performed for the RLC model. For each RDL wirelength, simulation is also performed for the RC model, where the same resistance and capacitance values as in the RLC model are used, but the inductance is not included.

As highlighted by the delay arc in Fig. 24(a), the 50% delay from the gate of the driver cell to the gate of the receiver cell is measured for both RC and RLC interconnect model simulations. Fig. 24(b) shows the plotted simulation data for INV_X16 of the Nangate45nm cell library. As evident from the figure, as the inductance increases with the wirelength, the RC model of the interconnect cannot accurately estimate the propagation delay. Fig. 24(c) shows the relative error of the RC interconnect model w.r.t the RLC model delay measured through SPICE simulation. As observed from this figure, the RC model can underestimate the propagation delay by approximately 30%. This result is consistent with previous studies [33, 31], confirming it is essential to include the inductance impact on timing in the design, analysis, and optimization steps of a system connected through RDL wires.

4.1.2 RLC Delay Model

To estimate the delay through RDL wires of a 2.5D system, a delay model is developed using SPICE simulations. On RDLs between the chiplets, point-to-point connections are more common. For that reason, a transmission line-based RLC delay model is used, as presented by Ismail and Friedman in [33]. Based on their work, using the transmission line model, the delay t_{pd} is a function of three variables, ζ, R_T, and C_T, as shown in (6).

$$R_T = \frac{R_{tr}}{R_t},\ C_T = \frac{C_L}{C_t},\ \zeta_{line} = \frac{R_t}{2}\sqrt{\frac{C_t}{Lt}} \tag{4}$$

$$\zeta = \zeta_{line}\frac{R_T + C_T + R_T C_T + 0.5}{\sqrt{1 + C_T}} \tag{5}$$

$$t_{pd} = \frac{t'_{pd}(\zeta, R_T, C_T)}{\omega_n} \tag{6}$$

They derived their final model for t_{pd} using only one variable, ζ, for the condition $0 < R_T, C_T < 1$. However, in this experimental setup, using Nangate45nm with InFO-like integration technology, this condition is not met. Additionally, to develop a more accurate delay model for each driver cell, two variables are selected, namely ζ_{line} and C_T. The selection is based on the fact that, as seen from (4) to (6), t_{pd} is dependent on R_{tr}, R_t, C_L, C_t, L_t. For a given driver, R_{tr} is fixed and considering ζ_{line} and C_T includes the rest of the variables.

$$scalingFactor = k + a\zeta_{line}^3 + b\zeta_{line}^2 + c\zeta_{line} + d\zeta_{line}^2 C_T \tag{7}$$

$$RLC\ Delay = scalingFactor \times RC\ Delay \tag{8}$$

For a given driver, using ζ_{line} and C_T, a factor using (7) is calculated. As shown in (8), this factor is multiplied with the RC delay to estimate the equivalent RLC delay for that driver. A multiplication factor is estimated because it is essential for modeling the RLC delay using RC parasitics as discussed in detail in the next section. Parameters of (7) are obtained by fitting the SPICE simulation data for each driver. Table 16 shows the fitted parameters for some cells of the

Nangate45nm cell library. Fig. 24(b) shows the delay estimated by the fitted model for the same cell side-by-side with the simulated RC and RLC models. As seen from Fig. 24(c), where the RC model underestimates the propagation delay by 30% for long wires, the fitted model estimates the delay with an error of only +/-1%. Using this RLC model, in the next section, a custom tool is developed, which can incorporate the RDL wire inductance impact on overall system performance in a holistic design flow, to iteratively optimize the chiplets to compensate for the overhead.

4.2 Holistic Co-Optimization Flow

In this section, the holistic iterative co-design flow is presented, which uses all standard libraries to design custom pin drivers taking into account the inductance overhead on overall system performance.

4.2.1 Overall Flow

Fig. 25 demonstrates the steps of the holistic iterative co-optimization flow. The gate-level netlist is synthesized from the RTL description of the entire 2.5D system. This gate-level netlist is then partitioned using a partition tool, which takes into account the impact of package wires on the timing while exploring the partition solutions. After the system is partitioned into chiplets, co-planning of chiplets and the package are performed together in a unified design environment. A unified PDK is designed using similar settings as presented in Table 6 to set up the unified environment. As a result, design information can be exchanged between chiplets and the package while performing timing budget extraction and hierarchical sub-design formation of the chiplets and the package. Using an RDL planning tool, the package RC loads are calculated based on the wirelength of the RDL nets. These estimated loads are appended with the hierarchical contexts, created by the top-level design tool, for each chiplets. After this step, chiplets and the package sub-designs can be implemented independently in parallel using the top-level constraints.

The traditional 2D chip design flow is followed to implement the chiplets. However, as the top-level constraints are always in effect during the design and optimization steps, the chiplet

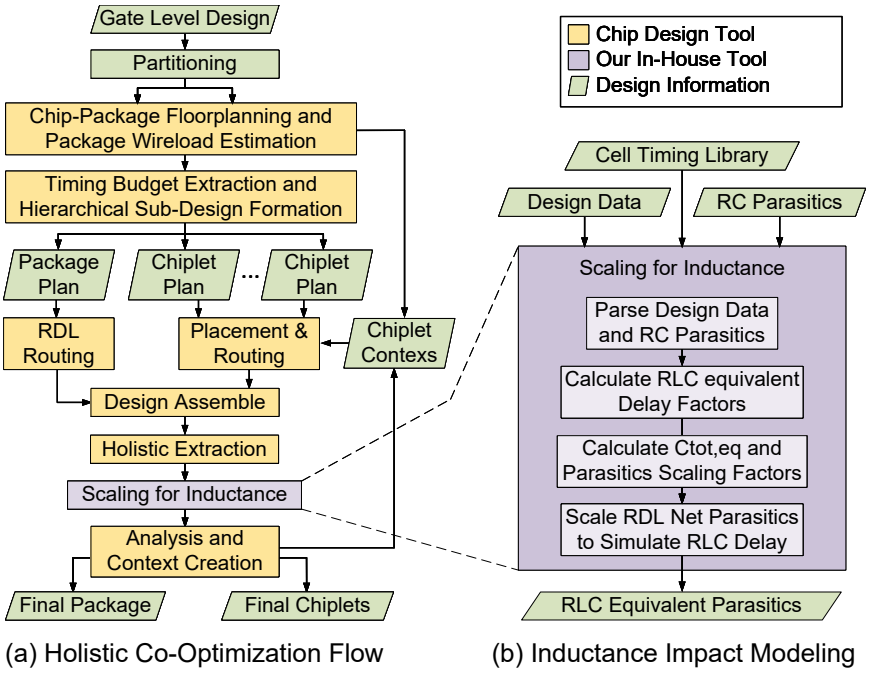


Figure 25: Holistic co-optimization flow with RDL inductance impact on timing

design tool takes into account the holistic considerations imposed in the top-level planning step. After finishing the chiplet designs, they are individually checked for design rule violations. If all the chiplets are violation-free, they are again assembled together in the "Design Assemble" step in the unified design environment for holistic extraction.

In the "Scaling for Inductance" step of this flow, a custom tool is utilized to modify the holistically-extracted parasitic netlist to include the inductance impact on the timing paths going through RDL wires. The inner workings of this tool are discussed in detail in Section 4.2.2. The tool adjusts the capacitances on the RDL nets in the parasitic netlist such that the STA and context creation tool calculates the RLC equivalent delay on the timing arcs going between chiplets through RDL wires. This adjusted parasitic netlist is used to perform timing analysis and create timing contexts for all chiplets using standard STA tools. As a result, the timing context created for each chiplet contains the timing overhead caused not only by the RC elements, but also the inductive element of the RDL wires.

These timing contexts of the chiplets are used to re-implement the chiplets in their own design environments, but with more accurate estimates of the timing constraints. As a result,

Table 16: Model parameters for selected Nangate45nm cells

Parameter	INV			BUF		
	X1	X4	X16	X1	X4	X16
a	-0.023	0.132	3.312	0.004	-0.036	1.931
b	0.047	-0.242	-5.783	0.007	0.000	-3.788
c	-0.013	0.156	2.804	-0.006	0.048	2.085
d	-0.008	-0.136	-1.009	-0.007	-0.076	-0.561
k	1.003	1.001	0.961	1.004	1.008	0.938

during timing optimization steps, the chip design tool makes adjustments within chiplets to compensate for the timing overhead caused by the RDL interconnects. This iterative optimization can be performed until the design goal is met or no more improvement can be achieved. Finally, the finished chiplets and the package designs can be analyzed and checked for the sign-off verifications.

4.2.2 Parasitics Scaling for Inductance

Though holistic extraction using existing industry-standard tools can accurately capture the capacitive coupling between chiplets and the package, it cannot capture the inductive impact of package wires. As discussed in Section 4.1, RDL wire inductance can severely affect the overall system performance. Existing commercial STA tools do not support inductance during timing analysis. Although standard parasitics description formats like SPEF support inductance, industry standard tools like Synopsys PrimeTime simply ignore those elements. That is why a custom tool is developed that can utilize an RLC delay model and adjust the parasitic information for existing STA tools in a way such that the timing contexts created for iterative optimization will take into account the timing overhead caused by the RDL inductance.

Fig. 25(b) shows the workflow of the parasitic adjustment tool. It reads the design data and parasitics netlist extracted through the holistic extraction process. The design data is parsed for RDL nets, their wirelength, driver, and receiver cells within the chiplet. This information is necessary to estimate the RLC equivalent delay using the model discussed in the previous section. The holistically-extracted parasitics contain the RC netlist of the entire system, both chiplets, and

package. This netlist is parsed for the RC netlist of the RDL wires. The tool also reads in the timing calculations performed by the analysis and context creation tool using the RC parasitics. All of this information is used to calculate the scaling factor, using equation (7) for driver cells and their RDL interconnects.

The delay calculation result extracted from the STA tool is used to calculate the total RC delay, from the driver input pin in one chiplet to the receiver input pin in another chiplet, by adding the driver cell-timing arc and the RDL wire net-timing arc. Using the parasitics and design information parsed in the previous step, the RDL interconnect parameters, ζ_{line} and C_T, are calculated. Then, using these interconnect parameters in equation (7), along with the fitted parameters for the driver, $scalingFactor$ is computed. The RC delay, calculated using the STA tool report, is multiplied by this $scalingFactor$ to estimate the RLC delay of the timing arc as shown in Fig 24(a). This estimated RLC delay is used to adjust the RC parasitics netlist and to allow the context creation tool to calculate RLC delay instead of RC delay for the paths running between the chiplets through RDL.

Standard STA tools calculate a path delay as a summation of timing arcs: gate delay from an input pin to its output pin is defined as the cell-arc, and the net delay from an output pin to the input pin is defined as the net-arc. The cell-arc is calculated using a look-up table (LUT) described in the cell timing library, and the net-arc is calculated using the Elmore-delay model on the RC tree of a net. The cell-arc depends on the input transition time (t_r) and the total capacitance (C_{tot}) at the output pin. The way Elmore-delay works on a RC tree, keeping the resistances of the RC tree constant, if a common factor scales all the capacitances in the tree, the total net-delay will be scaled by the same factor.

$$\begin{aligned} RLC &= cell\ delay + net\ delay \\ &= LUT(C_{tot,eq}, t_r) + scalePar \times (RC\ net\ delay) \end{aligned} \tag{9}$$

Where,

C_{tot} : Total Capacitance in the Parasitic RC network,

t_r : Input transition time of the driver cell,

$C_{tot,eq}$: Total equivalent capacitance in the parasitic network required to simulate the estimated RLC delay, and

LUT : Cell timing library look-up table

$scalePar$: $C_{tot,eq}/C_{tot}$

The delay estimated by the RLC model for a specific RDL interconnect, using equations (7)-(8), is the summation of the cell-arc of the driver gate and the net-arc of the RDL wire. Based on the aforementioned relationships, the equation (9) is developed, which can be used to look up the total equivalent capacitance ($C_{tot,eq}$) from the cell timing library. That capacitance, when used in the RC parasitic netlist, will force the STA tool to compute the RLC delays for the timing arc from the driver input to the receiver input pins of the RDL interconnects. As a result, the timing contexts for all chiplets created by the tool will take into consideration the timing overhead caused by the RDL wire inductance, along with the RC elements.

To adjust the RC parasitic netlist, a scaling factor, *scalePar*, is calculated for each RDL net. The factor *scalePar* is defined as the ratio of the RLC equivalent total capacitance ($C_{tot,eq}$) and the total capacitance (C_{tot}) on the RC tree of the extracted parasitics netlist. This *scalePar* factor is used to multiply all the capacitances attached to the RDL net RC tree in the parasitic netlist. This scaled parasitic netlist is exported to be used by the STA tool for timing analysis and chiplet timing-context creation. As only the RDL net capacitances are scaled, all the delays calculated by the STA tool using this scaled parasitic netlist, from RDL wire driver input to the receiver input, are the equivalent RLC delay, although the chiplet internal nets are still calculated using the RC delay models. Due to the iterative nature of the design flow, as shown in Fig. 25, the chiplet design tools then adjust the RDL wire drivers and receiver, as well as the internal gates of the chiplets to compensate for the timing overhead caused by the RDL interconnect. Thus, this flow achieves the co-design and optimization goals of the overall 2.5D system.

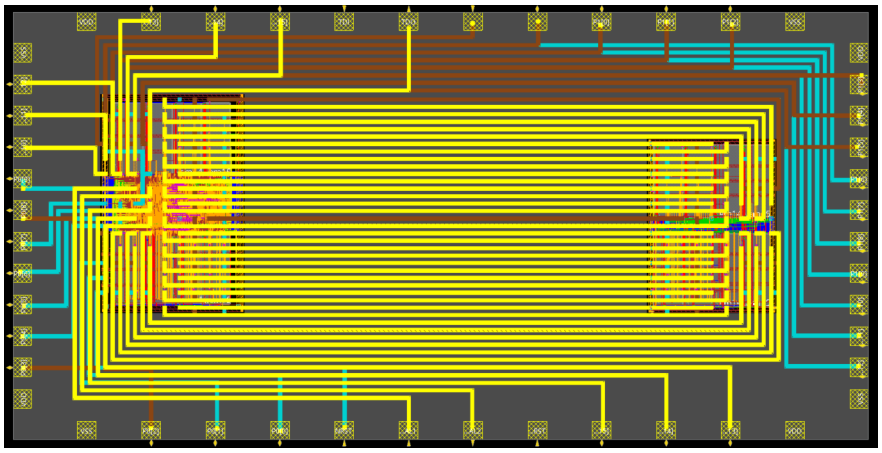
Due to the modular design of the adjustment tool, the RLC interconnect delay model is separate from the rest of the parts of the tool. A more accurate and physics-based model can replace this delay model to create a more accurate and general tool. Combined with existing ASIC CAD tools, they implement holistic co-optimization flows with accurate inductance considerations on timing.

4.3 Experimental Study

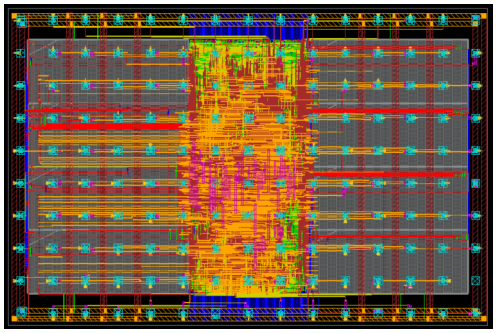
4.3.1 Design Setup

To study this flow on a real design, the microcontroller system is implemented using the 7M3R version of the Nangate45nm technology. Fig. 26(a) shows the package floorplan and RDL routing of the experimental system. This system, being a small one, can be designed using very short package wires. The chiplets are placed 1000 µm apart on the package, as it is typical for the chiplets to be connected with RDL wires in 1 cm range. As the chiplets are small, only the minimum spacing is used. There are 100 signals running between the two chiplets in this design, which have wirelength varying in the range 1000–2500 µm. In the SPICE simulation, this wirelength range is covered and the driver RLC delay parameters are fitted based on the simulation results.

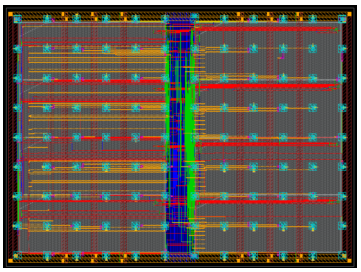
Following the holistic iterative co-optimization flow discussed in the previous section, the entire system is implemented at a target system frequency of 300 MHz. Fig. 26(b)-(c) shows the finished physical design of the two chiplets. Two different designs are prepared for comparative study. In one design, the exact flow of Fig. 25 is followed to perform iterative optimization through holistic extraction and parasitics netlist scaling for inductance consideration. In the other design, all the steps of this flow are followed except for the “Scaling for Inductance” step. The original parasitics extracted through holistic extraction is directly used for timing analysis, context creation, and iterative optimizations. This way, we can pinpoint the exact optimizations performed by the chip design tool accounting for the inductance impact on the overall system performance. Both of the designs required two iterations to reach their best performance.



(a) Assembled 2.5D system with chiplets and the package together



(b) Core-Chiplet



(c) Memory Chiplet

Figure 26: Physical design layouts of chiplets and the package

4.3.2 Analysis and Results

Here, the system with RDL inductance considerations using scaled parasitics is referred as the RLC-Design, and the system without RDL inductance considerations as the RC-Design. Fig. 27 shows the histogram of the timing analysis result of the paths going through the RDL wires in the RC-Design. The paths with total delay varying within 0.05 ns are binned together. The red bars and green bars show the analysis result obtained using the holistically-extracted RC parasitics and RLC equivalent scaled parasitics, respectively. As seen from the figure, without consideration of the inductive overhead on the timing path, the STA tool reports zero violating paths at the target frequency of 300 MHz. However, when STA is performed on the same design taking into account the inductive overhead, approximately 35% of the paths through the package violate the timing requirement. The worst violating paths miss the timing requirement by 0.15 ns. In a

high-performance GHz design, that means a violation by 20–30% of the clock period. Without careful considerations of these timing overhead caused by the inductive behavior of the RDL wires, the system will fail to run at its nominal speed, even though the sign-off verification report says the system met the timing requirement.

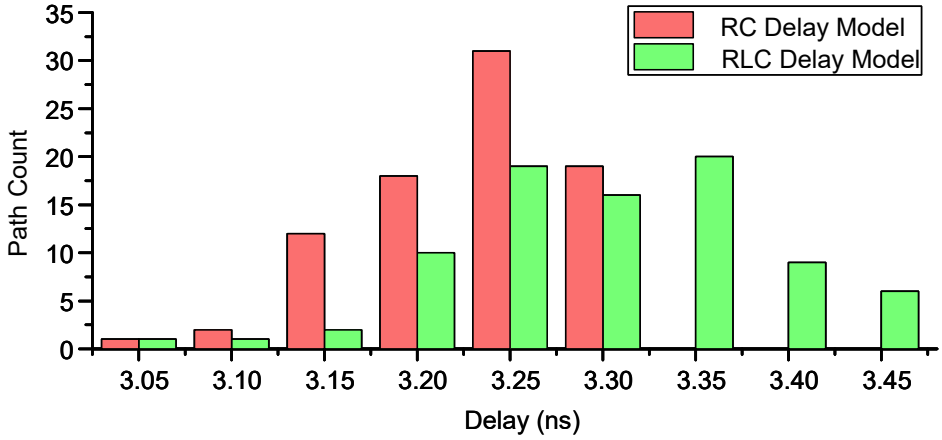


Figure 27: Timing path count per 0.05 ns delay bin through RDL

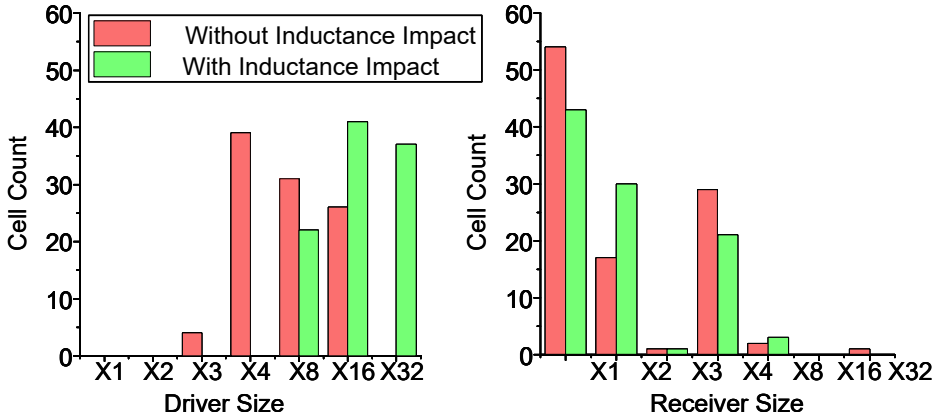


Figure 28: Package inductance impact on cell size distribution

After every iteration of the chiplet physical design, as shown in Fig 25, finished chiplets are assembled with the package for holistic extraction, scaling for inductance, and context creation. Using the chiplet timing context created after the previous iteration means the chiplet physical design tool obtains a more accurate view of the overall system in every subsequent iteration. As a result, it can fine-tune the chiplet designs to suit the system requirements. In the experimental designs, a significant difference is observed in the cell sizes of timing paths going through RDL

wires. Fig. 28 shows the distribution of cell size of the drivers and receivers of the signals going through RDL wires. The red and green bars show the cell counts of the final implementations of the RC-Design and the RLC-Design, respectively. As observed from the driver size distribution, the chiplet physical design tool has inserted larger drivers in the RLC-Design to compensate for the delay overhead caused by the inductive effect of RDL wires. Unaware of this delay overhead, the RC-Design uses drivers as small as X3, thus failing to meet the timing requirement. As the only difference between the two implementations is the consideration of RDL wire inductance, this shift in the distribution of the driver size clearly highlights that the chiplet design tool successfully optimizes the delay impact of the inductance of RDL wires.

Table 17: Changes in receivers between RC and RLC Designs

Design	Path-1	Path-2	Path-3
RC	BUF_X4 BUF_X1 BUF_X8 BUF_X2	AOI21_X1 NAND4_X1 XOR2_X1 BUF_X1	AOI22_X4
RLC	BUF_X2	BUF_X1	AOI22_X2

Significant changes are also observed on the receiver side of RDL wires. The size distribution of the receiver cells in Fig. 28 shows that many of the larger receiver cells in the RC-Design are replaced by smaller cells in the RLC-Design. For example, although there are many X4 and some X32 receivers in the RC-Design, they are replaced by smaller X2 cells in the RLC-Design. This change is performed by the chiplet design tool to reduce the capacitive load on the chiplet pin, which reduces the overall delay on the affected paths.

When using even a large cell in the driving chiplet is not enough to meet the timing, the chiplet design tool takes drastic measures on the receiver side, which is briefly highlighted in Table 17. Though, in general, the receivers are downsized to reduce the load at the input pins, it can be noticed in Fig. 28 that the X1 receiver count decreased and the X2 receiver count increased in the RLC-design. These changes are due to the adjustments performed as in Path-1 and Path-2 of Table 17. In Path-1, four buffer cells of different sizes in the RC-Design are replaced by only one buffer of size X2. In Path-2 of RC-Design, different logic cells of size X1 are directly

connected to the chiplet input pin putting a large capacitive load on it. In the optimization steps of the RLC-Design, a single X1 buffer is placed as the receiver that subsequently drives the logic cells within the receiver chiplet. That is, a group of smaller X1 receivers connected to a single input pin is replaced by a single X1 or X2 receiver, thus decreasing overall X1 receiver count or increasing X2 receiver count. In some cases, as in Path-3, where inserting a buffer as the receiver does not help in timing improvement, it simply reduced the logic gate cell size. Note that both chiplets are implemented independently in parallel in their own design environment. This cross-boundary co-optimization between chiplets, to compensate for the inductive delay overhead, is achieved by using chiplet timing contexts created using the RLC equivalent parasitics generated by the custom adjustment tool.

Chapter 5

Conclusion and Future Work

In conclusion, this thesis focuses on developing several tool flows and methodologies for design, extraction, analysis, and optimization of high-performance 2.5D systems. Through several design case studies, it is shown that chiplet-package interactions in high-density integration technologies significantly affect the overall system performance. The traditional die-by-die approach is not sufficiently accurate for advanced packaging solutions. The holistic extraction flow can accurately capture these cross-boundary interactions and optimize the system to reduce the overhead of package wires on system performance. Moreover, the flow supports several agile design approaches that can be exploited to generate different flavors of a 2.5D system with almost zero design cost, as illustrated using a three-way partitioned system. These holistic design methodologies offer designers and application engineers flexibility, low-cost customization, and performance. This flow is tested in silicon and can be utilized to implement 2.5D systems in commercial technologies using standard VLSI CAD tools.

While the holistic flow can accurately capture interactions between all components and perform system-level optimization, it is not very scalable. Moreover, it cannot accommodate heterogeneous technologies. The proposed in-context flows overcome these limitations by decoupling the design steps involving heterogeneity, while maintaining a holistic view at planning, analysis, and optimization steps. The different flavors of the in-context flow offers trade-offs between scalability and accuracy. Through comparative studies between several implementations of a homogeneous system, it is shown that the in-context extraction methodologies can achieve holistic-like accuracy. It is also shown that the proposed flows can effectively optimize heterogeneous 2.5D systems and achieve holistic-like performance.

The inductance-aware chiplet-package co-optimization flow takes into account the delay overhead caused by the inductance and RC elements of the RDL wires. The proposed RLC delay model can achieve SPICE-like analysis accuracy. However, this delay model can be replaced with

a more accurate physics-based model. Through the use of RLC equivalent parasitics and iterative optimization, this flow can automatically co-optimize the drivers and receivers, in different chiplets, connected to the same RDL wire keeping the chiplet physical design process parallel and independent.

Though all the tool flows and methodologies presented here are focused on timing optimization, they are general frameworks for cross-boundary design, analysis, and optimizations. Just like the timing context creation in future work, similar contexts can be extracted for other design parameters like IR-drop, thermal, signal integrity, etc. These contexts can be converted to design constraints and utilized in the iterative optimization of the system using the proposed flows. Except the holistic flow, all other flows can be utilized to design homogeneous as well as heterogeneous systems. All these flows being compatible with each other, can be combined to develop a unified flow that can offer their unique features at appropriate design steps. For example, the per-chiplet in-context flow can be utilized in the initial iterations to get a near optimal design of a homogeneous or a heterogeneous system system. This design can be further optimized with holistic or more accurate flavor of the in-context flows. The other detailed analysis steps, such as power integrity and signal integrity, thermal analyses, etc. are not performed in this study. Those steps can be included in the future versions of the flows. The parasitics adjustment tool enables the inductance delay overhead consideration in the existing STA tools and can be further extended to accommodate all RCLM elements in future work.

References

[1] S. Dwarakanath, P. M. Raj, A. Agarwal, D. Okamoto, A. Kubo, F. Liu, M. Kathaperumal, and R. R. Tummala, "Evaluation of Fine-Pitch Routing Capabilities of Advanced Dielectric Materials for High Speed Panel-RDL in 2.5D Interposer and Fan-Out Packages," in *IEEE Electronic Components and Technology Conference*, 2019, pp. 718–725.

[2] C.-C. Hsieh, C.-H. Wu, and D. Yu, "Analysis and Comparison of Thermal Performance of Advanced Packaging Technologies for State-of-the-Art Mobile Applications," in *IEEE Electronic Components and Technology Conference*, May 2016, pp. 1430–1438.

[3] S. Naffziger, K. Lepak, M. Paraschou, and M. Subramony, "2.2 AMD Chiplet Architecture for High-Performance Server and Desktop Products," in *IEEE International Solid-State Circuits Conference*, Feb. 2020, pp. 44–45.

[4] J. H. Lau, M. Li, M. L. Qingqian, T. Chen, I. Xu, Q. X. Yong, Z. Cheng, N. Fan, E. Kuah, Z. Li *et al.*, "Fan-out wafer-level packaging for heterogeneous integration," *IEEE Transactions on Components, Packaging, and Manufacturing Technology*, vol. 8, no. 9, pp. 1544–1560, Sep. 2018.

[5] S. Dwarakanath, P. M. Raj, A. Agarwal, D. Okamoto, A. Kubo, F. Liu, M. Kathaperumal, and R. R. Tummala, "Evaluation of Fine-Pitch Routing Capabilities of Advanced Dielectric Materials for High Speed Panel-RDL in 2.5 D Interposer and Fan-Out Packages," in *IEEE Electronic Components and Technology Conference*, May 2019, pp. 718–725.

[6] Y. Xie, C. Bao, and A. Srivastava, "Security-Aware 2.5D Integrated Circuit Design Flow Against Hardware IP Piracy," *Computer*, vol. 50, no. 5, pp. 62–71, May 2017.

[7] S. Patnaik, M. Ashraf, O. Sinanoglu, and J. Knechtel, "Best of Both Worlds: Integration of Split Manufacturing and Camouflaging into a Security-Driven CAD Flow for 3D ICs," in *International Conference on Computer-Aided Design*, Mar. 2018, pp. 1–8.

[8] J. H. Lau, "Recent advances and trends in fan-out wafer/panel-level packaging," *Journal of Electronic Packaging*, vol. 141, no. 4, p. 040801, 2019.

[9] M. Brunnbauer, T. Meyer, G. Ofner, K. Mueller, and R. Hagen, "Embedded Wafer Level Ball Grid Array (eWLB)," in *International Electronics Manufacturing Technology Conference*, Nov. 2008, pp. 1–6.

[10] W. Ki, W. Lee, I. MoK, I. Lee, W. Do, M. Kolbehdari, A. Copia, S. Jayaraman, C. Zwenger, and K. Lee, "Chip Stackable, Ultra-thin, High-Flexibility 3D FOWLP (3D SWIFT® Technology) for Hetero-Integrated Advanced 3D WL-SiP," in *IEEE Electronic Components and Technology Conference*, May 2018, pp. 580–586.

[11] C. Tseng, C. Liu, C. Wu, and D. Yu, "InFO (Wafer Level Integrated Fan-Out) Technology," in *IEEE Electronic Components and Technology Conference*, May 2016, pp. 1–6.

[12] J. Kim, G. Murali, H. Park, E. Qin, H. Kwon, V. C. K. Chekuri, N. M. Rahman, N. Dasari, A. Singh, M. Lee *et al.*, "Architecture, Chip, and Package Codesign Flow for Interposer-Based 2.5-D Chiplet Integration Enabling Heterogeneous IP Reuse," *IEEE Transactions on Very Large Scale Integration Systems*, vol. 28, no. 11, pp. 2424–2437, 2020.

[13] W. Liu, Min-Sheng Chang, and T. Wang, "Floorplanning and signal assignment for silicon interposer-based 3D ICs," in *Design Automation Conference*, June 2014, pp. 1–6.

[14] D. Yu, "A new integration technology platform: Integrated fan-out wafer-level-packaging for mobile applications," in *Symposium on VLSI Technology*, June 2015, pp. T46–T47.

[15] N. Chen, T. Hsieh, J. Jinn, P. Chang, F. Huang, J. Xiao, A. Chou, and B. Lin, "A Novel System in Package with Fan-Out WLP for High Speed SERDES Application," in *IEEE Electronic Components and Technology Conference*, May 2016, pp. 1495–1501.

[16] H. Pu, H. J. Kuo, C. S. Liu, and D. C. H. Yu, "A Novel Submicron Polymer Re-Distribution Layer Technology for Advanced InFO Packaging," in *IEEE Electronic Components and Technology Conference*, May 2018, pp. 45–51.

[17] C.-T. Wang, J.-S. Hsieh, V. C. Y. Chang, S. Huang, T. Ko, H. Pu, and D. Yu, "Signal Integrity of Submicron InFO Heterogeneous Integration for High Performance Computing Applications," in *IEEE Electronic Components and Technology Conference*, May 2019, pp. 688–694.

[18] H. Park, J. Kim, V. C. K. Chekuri, M. A. Dolatsara, M. Nabeel, A. Bojesomo, S. Patnaik, O. Sinanoglu, M. Swaminathan, S. Mukhopadhyay *et al.*, "Design Flow for Active Interposer-Based 2.5-D ICs and Study of RISC-V Architecture With Secure NoC," *IEEE Transactions on Components, Packaging and Manufacturing Technology*, vol. 10, no. 12, pp. 2047–2060, 2020.

[19] C.-H. Chiang, F.-Y. Chuang, and Y.-W. Chang, "Unified Redistribution Layer Routing for 2.5 D IC Packages," in *Asia and South Pacific Design Automation Conference*, Jan. 2020, pp. 331–337.

[20] H.-T. Wen, Y.-J. Cai, Y. Hsu, and Y.-W. Chang, "Via-based Redistribution Layer Routing for InFO Packages with Irregular Pad Structures," in *Design Automation Conference*, Jul. 2020, pp. 1–6.

[21] J. Kim, V. C. K. Chekuri, N. M. Rahman, M. A. Dolatsara, H. M. Torun, M. Swaminathan, S. Mukhopadhyay, and S. K. Lim, "Chiplet/Interposer Co-Design for Power Delivery Network Optimization in Heterogeneous 2.5D ICs," *IEEE Transactions on Components, Packaging, and Manufacturing Technology*, pp. 1–1, 2021.

[22] Y. Ma, L. Delshadtehrani, C. Demirkiran, J. L. Abellán, and A. Joshi, "TAP-2.5 D: A Thermally-Aware Chiplet Placement Methodology for 2.5 D Systems," in *Design, Automation and Test in Europe*. IEEE, Feb. 2021, pp. 1–6.

[23] P. Vivet, E. Guthmuller, Y. Thonnart, G. Pillonnet, G. Moritz, I. Miro-Panadès, C. Fuguet, J. Durupt, C. Bernard, D. Varreau *et al.*, "2.3 A 220GOPS 96-Core Processor with 6 Chiplets 3D-Stacked on an Active Interposer Offering 0.6 ns/mm Latency, 3Tb/s/mm^2 Inter-Chiplet Interconnects and 156mW/mm^2 @ 82%-Peak-Efficiency DC-DC Converters," in *IEEE International Solid-State Circuits Conference*, Feb. 2020, pp. 46–48.

[24] J. Wang, S. Ma, P. D. S. Manoj, M. Yu, R. Weerasekera, and H. Yu, "High-Speed and Low-Power 2.5D I/O Circuits for Memory-logic-integration by Through-Silicon Interposer," in *IEEE International 3D Systems Integration Conference*, Oct 2013, pp. 1–4.

[25] D. Xu, P. D. S. Manoj, H. Huang, N. Yu, and H. Yu, "An Energy-efficient 2.5D Through-silicon Interposer I/O with Self-adaptive Adjustment of Output-voltage Swing," in *International Symposium on Low Power Electronics and Design*, Aug. 2014, pp. 93–98.

[26] G. Karypis and V. Kumar, "Multilevel k-way Hypergraph Partitioning," *VLSI Design*, vol. 11, no. 3, pp. 285–300, 2000.

[27] J. Cong, S. K. Lim, and C. Wu, "Performance Driven Multi-level and Multiway Partitioning with Retiming," in *Design Automation Conference*, June 2000, pp. 274–279.

[28] T.-C. Lin, C.-C. Chi, and Y.-W. Chang, "Redistribution Layer Routing for Wafer-Level Integrated Fan-Out Package-on-Packages," in *International Conference on Computer-Aided Design*, Nov. 2017, pp. 561–568.

[29] M. R. Guthaus, J. E. Stine, S. Ataei, B. Chen, B. Wu, and M. Sarwar, "OpenRAM: An Open-source Memory Compiler," in *International Conference on Computer-Aided Design*, Nov 2016, pp. 93:1–93:6.

[30] Y. Peng, D. Petranovic, and S. K. Lim, "Chip/Package Co-Analysis and Inductance Extraction for Fan-Out Wafer-Level-Packaging," in *IEEE Conference on Electrical Performance of Electronic Packaging and Systems*, Oct. 2017, pp. 1–3.

[31] A. Shebaita, D. Petranovic, and Y. Ismail, "Including Inductance in Static Timing Analysis," 2007, pp. 686–691.

[32] A. C. Yucel, I. P. Georgakis, A. G. Polimeridis, H. Bağcı, and J. K. White, "VoxHenry: FFT-Accelerated Inductance Extraction for Voxelized Geometries," vol. 66, no. 4, pp. 1723–1735, 2018.

[33] Y. I. Ismail and E. G. Friedman, "Effects of Inductance on the Propagation Delay and Repeater Insertion in VLSI Circuits," *IEEE Transactions on Very Large Scale Integration Systems*, vol. 8, no. 2, pp. 195–206, 2000.

[34] H. A. Aebischer and B. Aebischer, "Improved Formulae for the Inductance of Straight Wires," *Advanced Electromagnetics*, vol. 3, no. 1, pp. 31–43, 2014.

Appendix

List of Published Papers

Chapter 2 through Chapter 4 are partial reproductions of papers that have been published or considered for publication at the following outlets:

Chapter 2:

> MD Arafat Kabir, and Yarui Peng, "Holistic Chiplet-Package Co-Optimization for Agile Custom 2.5D Design", IEEE Transactions on Components, Packaging, and Manufacturing Technology, vol. 11, no. 5, pp. 715–726, 2021.
>
> The above journal article is an extension of the following published conference papers:
>
> > MD Arafat Kabir, and Yarui Peng, "Holistic 2.5D Chiplet Design Flow: A 65nm Shared-Block Microcontroller Case Study", in Proc. IEEE International System-on-Chip Conference, pp. 277-282, Oct 2020.
> >
> > MD Arafat Kabir, and Yarui Peng, "Chiplet-Package Co-Design For 2.5D Systems Using Standard ASIC CAD Tools", in Proc. Asia and South Pacific Design Automation Conference, pp. 351–356, Jan 2020.

Chapter 3.3

> MD Arafat Kabir, Dusan Petranovic, and Yarui Peng, "Coupling Extraction and Optimization for Heterogeneous 2.5D Chiplet-Package Co-Design", in Proc. International Conference on Computer-Aided Design, pp. 1–8, Nov 2020.

Chapter 3.4

> MD Arafat Kabir, Weishiun Hung, Tsung-Yi Ho, and Yarui Peng, "Holistic and In-Context Design Flow for 2.5D Chiplet-Package Interaction Co-Optimization", in Proc. International Symposium on VLSI Design, Automation and Test, pp. 1–4, May 2021.

Chapter 3.5

> MD Arafat Kabir, Dusan Petranovic, and Yarui Peng, "A Scalable In-Context Design and Extraction Flow for Heterogeneous 2.5D Chiplet-Package Co-Optimization", in Proc. IEEE Conference on Electrical Performance of Electronic Packaging and Systems, Oct 2021.

Chapter 4

MD Arafat Kabir, Dusan Petranovic, and Yarui Peng, “Cross-Boundary Inductive Timing Optimization for 2.5D Chiplet-Package Co-Design”, in Proc. ACM Great Lakes Symposium on VLSI, pp. 135–140, Jun 2021.