Holistic and In-Context Design Flow for 2.5D Chiplet-Package Interaction Co-Optimization

MD Arafat Kabir[1], Weishiun Hung[2], Tsung-Yi Ho[2], and Yarui Peng[1]
[1]CSCE Department, [2]CS Department
[1]University of Arkansas, [2]National Tsing Hua University

***Abstract*—In recent days, 2.5D package designs have gained popularity with an increasing number of heterogeneous chiplets integrated into advanced system-in-packages. RDL wires become longer and denser, presenting a growing impact on system performance, reliability, and integrity. At present, there exists no standard CAD flow that can design, analyze, and optimize a complete heterogeneous 2.5D system. The traditional die-by-die design approach processes each component independently during extraction and optimization and cannot be applied to heterogeneous systems without fundamental changes in standard CAD tools. Not only the chiplet-package extraction is inaccurate between the die-package interface ignoring all RDL capacitive and inductive impacts, but traditional CAD tools are also unable to perform cross-boundary design optimization.**

We present a complete chiplet-package co-optimization flow for both homogeneous and heterogeneous 2.5D designs. It encompasses 2.5D-aware partitioning, chiplet-package co-planning, holistic and in-context extraction, package inductance consideration, and iterative optimization, along with design analysis and verification of the entire 2.5D system. In our previous work [1] targeting heterogeneous systems, we achieved an extraction error ranging between -2.10% and 24.0%. The in-context design flow proposed in this work achieves less than 1% extraction error on ground and coupling capacitance. This extraction result can be used to perform timing analysis with 99.8% accuracy and to generate timing context with 99.4% accuracy for iterative optimization.

***Keywords*—2.5D Design, Chiplet-Package Co-Optimization, Holistic, Heterogeneous, In-Context.**

I. Introduction

The demand for increased functionality and performance in modern chips is ever-growing. In recent days, 2.5D integration is providing many design solutions within a compact package. Apart from reduced package size, 2.5D packages have lower power, higher bandwidth, higher yield, and better thermal dissipation. It reduces the turn-around time through plug-and-play design techniques [2]. 2.5D integration has applications in IP-protection and hardware security. Heterogeneous integration also becomes one of the most attractive technologies, where chiplets from different technologies can be used in the same system.

Traditionally, 2.5D systems are designed in a die-by-die approach, where each chiplet is designed and optimized individually for the best performance. Then, these chiplets are integrated through redistribution layers (RDL) in a 2.5D package. Though this approach enables IP-reuse [3] and reduces the turn-around time, it cannot guarantee the best performance out of the system. A large performance margin needs to be left-out for the package overhead to ensure that the system can run reliably at the rated speed. As the extraction and optimizations of each component are performed independently, the interaction between chiplets and the package is completely ignored. Though possible, it is often not practical to perform cross-boundary optimizations between the package and chiplets. In high-density packages like InFO [4], the coupling between chiplets and the package is prominent and needs to be considered carefully in the extraction and optimization steps. Moreover, to achieve the best system performance, the entire system must be considered holistically with cross-boundary optimization flow.

This material is based upon work supported by the National Science Foundation under Grant No. 1755981. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

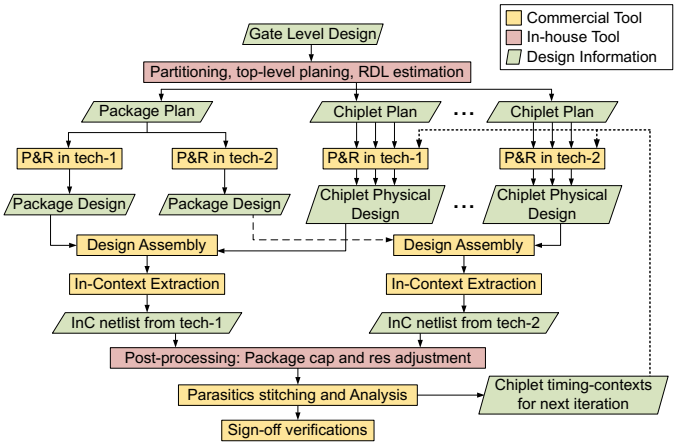


Fig. 1. Proposed in-context co-optimization flow for heterogeneous 2.5D systems

Integrating several chiplets into the same package will inevitably sacrifice individual chiplet's performance, say maximum operating frequency, often because the RDL wires will dominate the critical timing path. Sophisticated design flow and algorithms are desired to cope with the placement and routing of the system. Therefore, it is indispensable to co-optimize chiplets and their package to ensure the highest performance. Several important factors, including wirelength minimization and timing budget reduction, need to be considered to achieve the best system-level performance.

In this paper, we present our chiplet-package co-optimization flow for heterogeneous 2.5D systems. Our flow incorporates the features essential to perform accurate design, extraction, analysis, and optimization of the entire 2.5D system. In a holistic flow, a 2.5D system is analyzed as a complete system, chiplets and the package together. Fig. 1 illustrates our proposed flow for heterogeneous 2.5D systems. We employ an accurate in-context extraction strategy and perform adjustments using our in-house tools to create a holistic view of the system parasitics. We use these in-context parasitics to perform cross-boundary analysis and generate post-physical design timing contexts of all chiplets. Using these timing contexts, we perform iterative cross-boundary optimizations to ensure the best system-level performance. An in-context extraction strategy was first presented in [1] that can handle heterogeneous 2.5D systems. However, the extraction result was not accurate enough to ensure reliable analysis and timing context accuracy. In this work, our new in-context extraction method can achieve holistic-like accuracy and generate highly reliable analysis results and timing contexts for

iterative optimizations. We claim the following new contributions: (1) A revised extraction strategy to perform in-context extraction of heterogeneous 2.5D systems; (2) A new post-processing method to improve the accuracy of extraction and analysis results; (3) A comparative study to validate the effectiveness of our methodology.

In addition, we discuss the potential approaches to consider the impact of package inductance on system performance and iterative optimization of the package design. To our best knowledge, there exists no other tool flow that implements an in-context co-optimization flow for heterogeneous 2.5D systems with holistic-like accuracy and effectiveness in extraction, analysis, and optimizations.

II. Chiplet-Package Co-Design Flow

Our in-context flow for heterogeneous 2.5D systems is demonstrated in Fig. 1. 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.

A. Top-Level Planning

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. We use our in-house tools to prepare this top-level plan. Using this plan, we prepare a hierarchical design, where the package is treated as the top-level design with chiplets as sub-designs. After this hierarchical sub-design formation, chiplets and the package can be implemented independently in parallel.

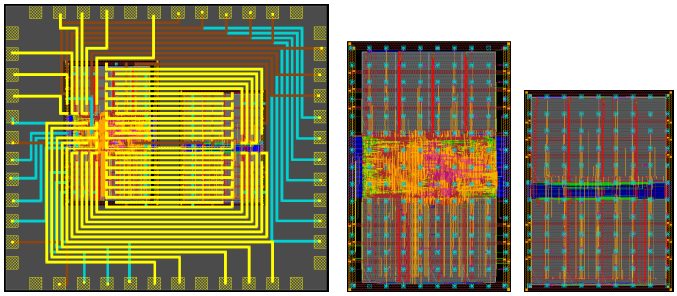
B. Physical Design

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. 1, 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 our in-house tool employing a greedy strategy.

C. In-Context Extraction and Post-Processing

After the physical design of chiplets, Design Rule Checking (DRC) is performed on them. Then, they are assembled with the package design for extraction. We perform in-context extraction per technology to capture the cross-boundary interactions among chiplets and the package. As presented in Fig. 1, 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, we have 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 our experimental study, we extract the top-level package parasitics treating all chiplets as blackbox macros. Our 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, we develop our in-house tool that can adjust the in-context parasitic netlist based on the top-level package parasitics.



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

Fig. 2. Chiplets and assembled package layouts of the homogeneous 2.5D system

Our 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 III.

$$layerFact_x = \frac{CapRDL_x - userFact \times TCapRDL_x}{CapRDL_x} \quad (1)$$

$$newNodeCap = nodeCap \times layerFact_x \quad (2)$$

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

D. Iterative Optimization

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. We use these contexts to re-implement the chiplets. This iterative approach is highlighted using the dotted line on the right-half of Fig. 1. Several iterations of chiplet physical design, assembly, extraction can be performed to optimize the system performance.

III. Experimental Study

A. Design Settings

In our experimental setup, we use an ARM Cortex-M0-based microcontroller system with two chiplets and 16KB memory, as presented in our previous work [1]. The Core-Chiplet contains all the

TABLE I
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

logic cells and 8KB memory, while the Mem-Chiplet contains the rest 8KB memory. We use a modified version of the Nangate 45nm PDK to implement the chiplets and the package, with the same settings as in Table 1 of [1]. The lower seven metal layers are used with original settings to implement the chiplets. The top three layers are modified to mimic the attributes of high-density 2.5D package RDLs. We use standard cells from the Nangate 45nm cell library. Memory macros are compiled using OpenRAM [5] memory compiler.

B. Homogeneous Holistic and In-Context Designs

For comparative study, we implement the homogeneous system using both holistic flow and our proposed in-context flow. The whole idea of the holistic flow is to treat a 2.5D system as a single design for analysis and optimization purposes. Such methods enable a globally-optimized system instead of a separated system containing components optimized within their individual domains but performing poorly once combined together. The details of our holistic flow can be found in our previous work [1].

In the first iteration of the chiplet physical design, the chiplets are implemented using top-level constraints and estimated package wireload. Fig. 2(b)–(c) shows these chiplets finished designs. In the holistic design, we assemble both chiplets with the package, as shown in Fig. 2(a), and perform holistic extraction. We use this holistic extraction method to perform iterative optimization of the chiplets. From here on, we refer to this design 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 II. These in-context parasitics are used in the iterative optimization of the system. From here on, we refer to this design as "Homogen-InC" design.

C. Analysis and Results

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 I 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 II presents the comparison between the holistic extraction and our proposed in-context extraction methodology results in this work and our previous work [1]. The extraction error in [1] varies between -2.10% to 24.0%. This much error cannot ensure reliable analysis results. In this work, the extraction error in both ground and coupling capacitance is less than 1%, which is a significant improvement over our previous work. In the parasitics adjustment tool, we use 0.4 with the Core-chiplet context and 0.6 with the Mem-chiplet context 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 II
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 (this work)	**0.00%**	**0.00%**	**0.01%**	**0.24%**	**0.6%**	**0.00%**
In-C GCAP Err in [1]	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 (this work)	**0.00%**	**0.04%**	**0.64%**	**0.03%**	**-0.01%**	**0.00%**
In-C CCAP Err in [1]	0.01%	0.17%	-2.10%	1.20%	2.81%	2.56%

TABLE III
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%

Table III 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 IV 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 our in-context flow is accurate and effective in designing high-performance heterogeneous 2.5D systems with very high reliability.

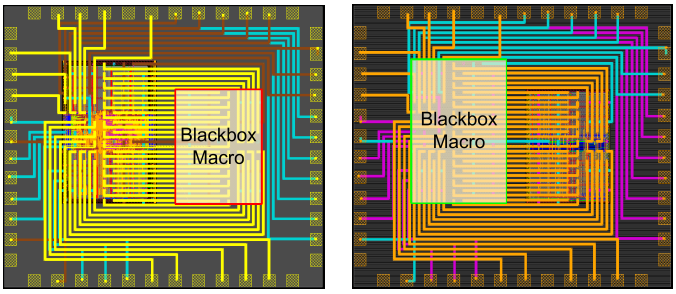
IV. HETEROGENEOUS DESIGN CASE-STUDY

A. Heterogeneous Design Setup

In this section, we present a design case-study of a heterogeneous system using 45nm technology. This is the same two-chiplet microcontroller system. However, in this design, the Mem-Chiplet is implemented using six metal layers and standard cells from the gscl45 cell library, which is bundled with the FreePDK45. Core-Chiplet is implemented the same as before. Though fundamentally, both chiplets are using the same device node, from the tool flow perspective, it is still a heterogeneous system. Since there are two chiplet routing stacks involved in this implementation, two unified technology stacks are generated for package design and assembly: one with seven chiplet routing layers and three RDLs (7M3R), and the other with six chiplet routing layers and three RDLs (6M3R). The routing layer parameters are the same as in Table 1 of [1].

B. In-Context Heterogeneous Design Results

After top-level planning, the first implementation of the chiplets is performed using timing contexts generated 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



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

Fig. 3. Layouts of the assembled heterogeneous system for in-context extraction

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. 3 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, we use the same *userFact* values in (1) as in the Homogen-InC design of Section III. 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 IV compares the performance of this design with the homogenous designs of Section III. 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 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 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 our flow can optimize heterogeneous 2.5D systems to achieve holistic-homogeneous system-like performance if that is feasible.

V. Further Optimizations

In our in-context flow, we tried to incorporate most critical parameters essential for chiplet-package co-optimization. However, some other parameters can be considered in the co-optimization steps. One such parameter is the package wire inductance. Traditionally, package inductance is modeled using S-parameters. This model cannot be directly used in the timing analysis. Moreover, standard timing analysis tools use only resistive (R) and capacitive (C) elements of the parasitics in the timing simulation. The tools simply ignore inductive (L) elements, even if they are specified in the parasitic netlist. As a result, some improvements in timing analysis and co-optimization tools are needed to consider inductance impact.

An immediate solution to include package inductance in the existing timing analysis flow is parasitics-scaling. After RC extraction, using an RLC package wireload model, the timing delay through package wires can be calculated separately. Using this delay, the RC values of the parasitic netlist can be adjusted to force the timing analysis tool to calculate the equivalent RLC delay using the RC parasitic netlist.

Our current RDL planner algorithm only routes wires between the chiplets. Package I/O pins are routed with a greedy algorithm. We will further design an algorithm to optimize package I/O routing with multi-objectives. The router will choose a proper escape boundary for each I/O pin by the relative position of its corresponding I/O to avoid possible detours. Routing from the boundary to I/O pads can be performed with wave propagation. Rip-up and reroute method will follow to reduce the minimum congestion. We will also consider integrating models such as URBER [6] or TAP 2.5D [7] to the current design flow in future work. This method can minimize the run-time and the wirelength during escape routing and jointly co-optimize thermal and performance together.

TABLE IV
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

VI. Conclusions

In this paper, we present our holistic and in-context design, extraction, analysis, and optimization flow for heterogeneous 2.5D systems. Through a comparative study between two implementations of a homogeneous system, we show that our in-context extraction methodology can achieve less than 1% error w.r.t holistic extraction method, which is a significant improvement from up to 24.0% error in our previous work [1]. This extraction method can achieve almost 100% accuracy in timing analysis and context generation. With a 45nm heterogeneous system combining two different PDKs, we show that our flow can optimize heterogeneous 2.5D systems and achieve holistic-like performance. Lastly, we discuss current limitations and further improvements of the flow, which will be explored in our future work.

References

[1] M. A. Kabir, D. Petranovic, and Y. Peng, "Extraction and Optimization for Heterogeneous 2.5D Chiplet-Package Co-Design," in *International Conference on Computer-Aided Design*, Nov. 2020, pp. 1–8.

[2] J. Kim, G. Murali, H. Park *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.

[3] H. Park, J. Kim, V. C. K. Chekuri *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.

[4] C. Wang, J. Hsieh, V. C. Y. Chang *et al.*, "Signal Integrity of Submicron InFO Heterogeneous Integration for High Performance Computing Applications," in *IEEE Electronic Components and Technology Conference*, May 2019, pp. 688–694.

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

[6] J. Weng, T.-Y. Ho, W. Ji *et al.*, "URBER: Ultrafast Rule-Based Escape Routing Method for Large-Scale Sample Delivery Biochips," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 39, no. 1, pp. 157–170, 2018.

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