2023 DESIGN AND VERIFICATION™ DVCON CONFERENCE AND EXHIBITION INDIA

Disaggregated methodology in Multi-die SoC– A Server SoC Case Study

Rohit Kumar Sinha
Bangalore 560035

Abstract- Server-SoCA is a multi-chiplet product with the compute die and base die being manufactured separately but connected on the platform through advanced packaging techniques. With growing power, performance, cost, and form factor challenges, the industry has been forced to look at alternatives beyond traditional monolithic solutions. One such approach is “disaggregation”, in which a design is broken into several “chiplets” which are connected using specialized die-to-die (D2D) IPs. There are no solutions (EDA tools, standards, methods) to create, optimize, edit, and manage interconnects for disaggregated systems. Although certainly not ideal, this has been tolerated because the number of interconnects to manage has been only a few thousand connections, a low enough count to manage through manual reviews and paranoia Complexity of interconnect and logic to support presents a complex problem to plan, maintain, generate logic and validate. Moreover, handling a server size disaggregated design and validating die to die async crossings required advanced methodologies and logic imposed for disaggregation requires extensive effort for all FE TFM activities.

In this paper, we are going to discuss what are the challenges associated with Server SoCs which includes multi-die chiplets and how to we handle multi-die voltage domain crossing, metastability and connectivity challenges. The paper will discuss about a methodology that can be majorly used for generation and optimization of 3D-IC or disaggregated system RTL and UPF.

I. Introduction

The number of transistors packed into a single chip has consistently increased over the years. Shrinking technology resulted in creating ever smaller transistors and resulted in increased Performance/Watt and reducing the Cost/Transistor. Some of the critical reason for disaggregated chiplet framework are

- With technology hitting physical limits, Moore's law is slowing
- Need a different approach to pack more transistors at lesser cost.
- The reticle limit and divide and conquer

- Silicon Packaging has undergone a sea change and is bringing the entire System on a Package.

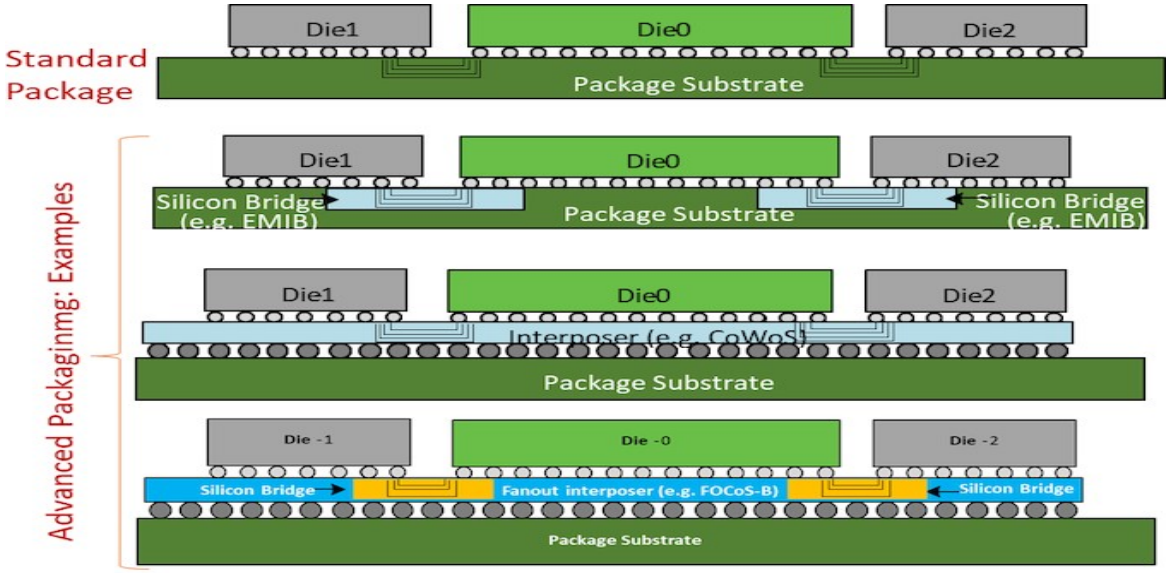
To understand the interconnect Disaggregation Methodology, it is important to define Interconnect Nomenclature. Disaggregation interconnects may be classified as:
(a) Die-Die Interconnects: Interconnects between stacked die that enable vertical interconnects between multiple die in a 3-D stack. These may be further sub-categorized using the process these interconnects are created with, which can lead to different physical attributes, such as Die-Die interconnects created using
a. Wafer-to-Wafer attach process
b. Die-to-Wafer attach process
c. Die-Die attach process
(b) On-package Die-Die Interconnects: For ex - 2D and Enhanced-2D Interconnects: Interconnects between die within the package that enable lateral connections.
(c) Die-to-Package Interconnects: Interconnects between the die and the package, typically known as the first level interconnect (FLI).
(d) Within-Package Interconnects: Interconnects within the package that enable lateral connections between two nodes or electrodes. Scaling projections of within-package interconnects are not discussed in this chapter. The reader is referred to the chapter on package substrate technologies
(e) Package-to-Board Interconnects: Interconnects between the package and the next level, which is typically the motherboard, are referred to as the second level interconnect (SLI)

Tile Vs Chiplet

- Tile is a SS or design hierachy that is instantiated multiple times to scale up the functionality.
    - For example, CPU tile can be placed multiple times to increase cores
    - In the server design, core tile spans across top die and base die.
- Chiplet is a die that attaches to the base die or main die.
    - For example, accelerator chiplet

Chiplets are already being used to mix dies from multiple chipmakers or from multiple process nodes, and they’re being used to build large chips that otherwise wouldn’t be possible due to reticle limits. All of which is being driven

by either economics in some fashion (not using an expensive, bleeding-edge node for every part of a chip), or a desire to combine IP from disparate manufacturers in a more expedient fashion than spending years taping out a monolithic chip. To be sure, monolithic chips as a whole aren't going away entirely (moving data remains expensive), but the economics of chip design are inexorably driving the use of chiplets in more cases.
Fig1 shows different packaging options that it is implemented in 2D or 2.5D ICs. Table1 explains that the comparison of standard and advance packaging options



(b. Packaging Options: 2D and 2.5D)

Figure 1

Characteristics / KPIs	Standard Package	Advanced Package	Comments
Characteristics			
Data Rate (GT/s)	4, 8, 12, 16, 24, 32		Lower speeds must be supported -interop (e.g., 4, 8, 12 for 12G device)
Width (each cluster)	16	64	Width degradation in Standard, spare lanes in Advanced
Bump Pitch (um)	100 – 130	25 - 55	Interoperate across bump pitches in each package type across nodes
Channel Reach (mm)	<= 25	<=2	
Target for Key Metrics			
B/W Shoreline (GB/s/mm)	28 – 224	165 – 1317	Conservatively estimated: AP: 45u for AP; Standard: 110u; Proportionate to data rate (4G – 32G)
B/W Density (GB/s/mm^2)	22-125	188-1350	
Power Efficiency target (pJ/b)	0.5	0.25	
Low-power entry/exit	0.5ns <=16G, 0.5-1ns >=24G		Power savings estimated at >= 85%
Latency (Tx + Rx)	< 2ns		Includes D2D Adapter and PHY (FDI to bump and back)
Reliability (FIT)	0 < FIT (Failure In Time) << 1		FIT: #failures in a billion hours (expecting ~1E-10) w/ CXi Flit Mode

Table 1

II. DISAGGREGATION FLOW

To address the growing scope and challenges of multiple chiplet dissgregation, a novel Compiler framework to create, optimize and manage chiplet interconnect using standard interfaces (raising the level of abstraction), to introduce industry automation, and to provide a robust methodology for generating correct-by-construction collateral (RTL, UPF and Validation collateral).

- Tool is a compiler framework built on EDA Solution tools that helps to create, optimize and manage chiplet interconnect
- It can be majorly used for generation and optimisation of 3D-IC or disaggregated system RTL and UPF.
- Tool was developed to ease the complex work on System level Pinlist, RTL, UPF creation and their alignment across dies.

- The goal of Tool is to centralise all required inputs under one tool that will generate top to bottom, correct by construction system level collaterals which is fully aligned between the dies.
- Tool can also supply a bump connection to the system dies, to ease D2Ds and IOs integration internal to the dies

Tool's solution consists of a) standard chiplet interfaces, b) system description and interconnection using high-level language primitives and c) a solver for design space exploration and D2D interconnect optimization. A standardized format is used to capture interface definitions for system level "Links" (example: CXL, DDR4/5, IDI, etc.). Parameters enable reuse of standard interfaces across product configurations in same family of products or across a family of products. Tool provides a rich set of high-level primitives developed using Python to succinctly describe system interconnections and configure their parameters (including power-domain and meta-attributes). This allows for higher levels of abstraction to be used to describe and inter-connect chiplets using standard interfaces with significantly reduced lines of code that are intuitive and correct-byconstruction. Tool primitives handle the Z dimension for interchiplet connections. This also allows for easy portability of standard interfaces across product configurations, and even across different product families to be readily re-used and (or) connected in different ways.
Tool Inputs and Outputs

Tool is built using EDA Solution's RTL Integration Platform and Python APIs. There are 3 primary inputs to Tool – Interface Spec (IFC SPEC), connectivity (CONN) and solver cost functions [optional]. The Tool engine takes primary inputs to generate the following outputs – RTL, UPF, Pinlist, Validation collaterals, Hand-off to Backend Collaterals. Below Table 2 explains different connections and nomenclature for the interconnects-

Chiplets/Dies/Tiles	Different dies that form the 3D-IC System.
Interface Specification (ifc_spec)	A file in a pre-defined ASCII or Python format used to define the specification of a standard protocol or ad-hoc protocol.
Link	A container that is used to define a collection of standard logical bundles or ad-hoc logical bundles.
	A link can contain one or more logical bundles.

	One or more logical bundles put together form the specification of a standard interface protocol or ad-hoc interface protocol for die-to-die interconnects.
Logical Bundle	A collection of ports specifying a particular logic function within a link.
Connections or Connectivity (conn)	Definition of how different chiplets in a 3D-IC system are connected using combination of links and (or) logical bundles.

Table 2

Figure 2 explains the flow diagram for Front End die2die and package2die validation and its framework. As mentioned in the picture, the flow requires 2 main inputs –

a) standard chiplet interfaces

b) system description and interconnection using high-level language instructions

Using the tool, we can generate the system level collaterals such as RTL, UPF, verification collaterals and synthesis collaterals.

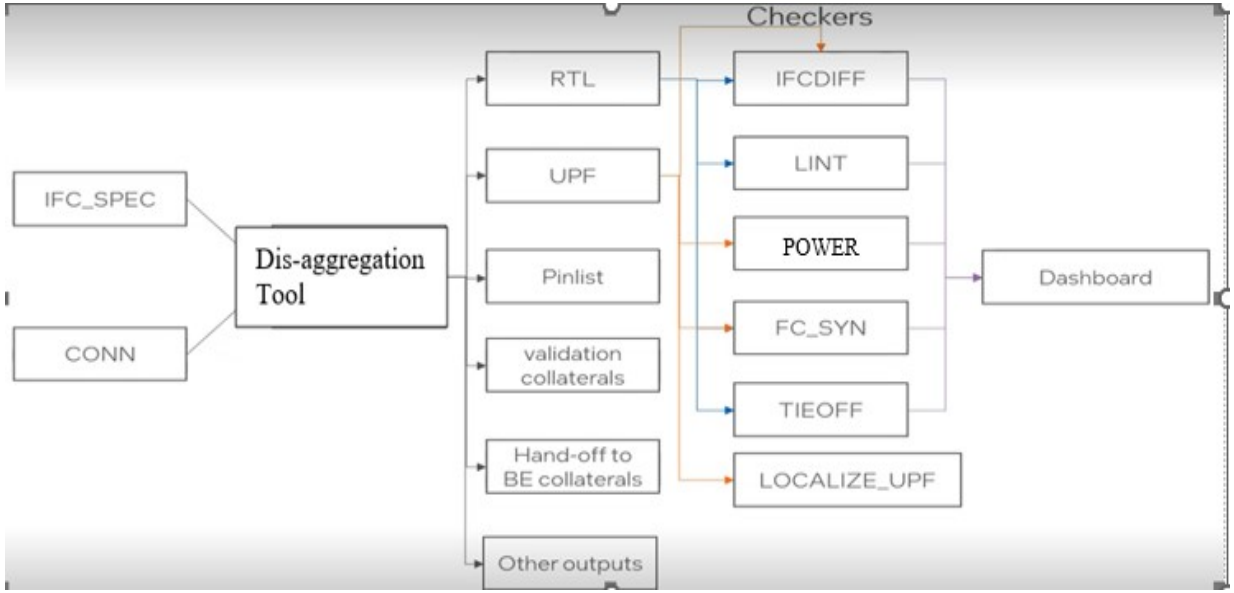

Checkers
RTL
UPF
Pinlist
validation collaterals
Hand-off to BE collaterals
Other outputs
IFC_SPEC
CONN
Dis-aggregation Tool
IFCDIFF
LINT
POWER
FC_SYN
TIEOFF
LOCALIZE_UPF
Dashboard

Figure 2

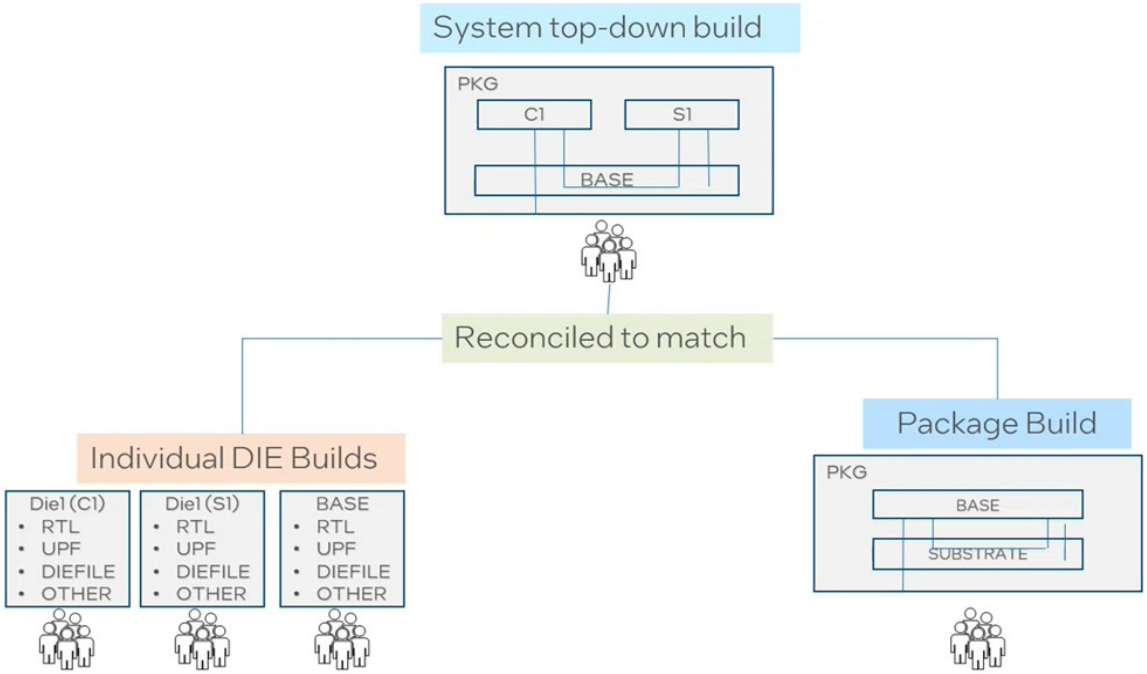
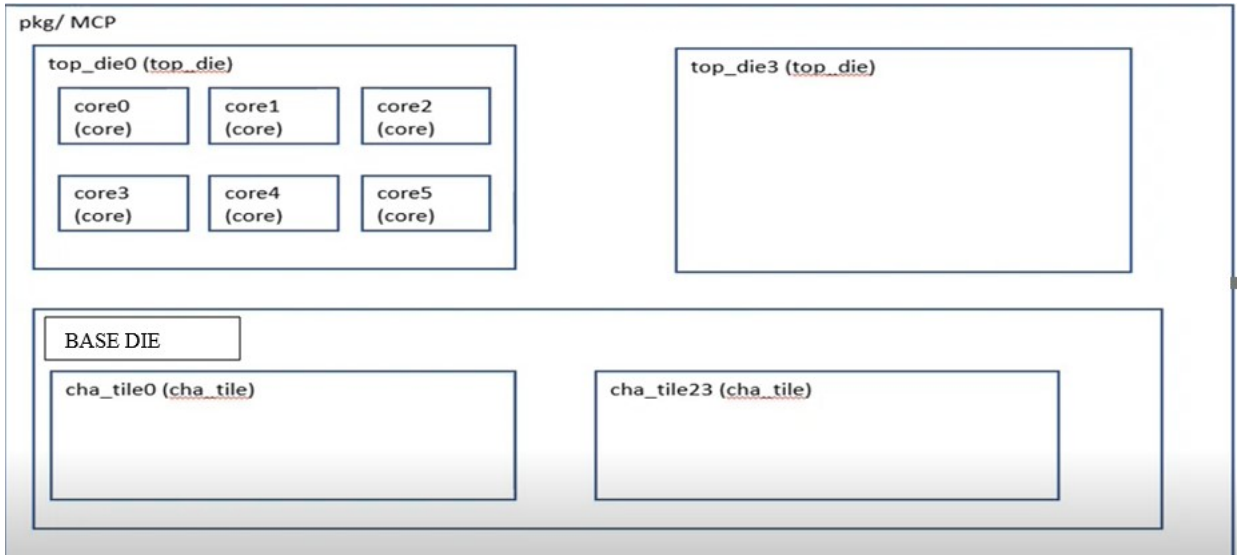

System top-down build
PKG
C1
S1
BASE
Reconciled to match
Individual DIE Builds
Die1 (C1)
RTL
UPF
DIEFILE
OTHER
Die1 (S1)
RTL
UPF
DIEFILE
OTHER
BASE
RTL
UPF
DIEFILE
OTHER
Package Build
PKG
BASE
SUBSTRATE

Figure 3


pkg/ MCP
top_die0 (top_die)
core0 (core)
core1 (core)
core2 (core)
core3 (core)
core4 (core)
core5 (core)
top_die3 (top_die)
BASE DIE
cha_tile0 (cha_tile)
cha_tile23 (cha_tile)

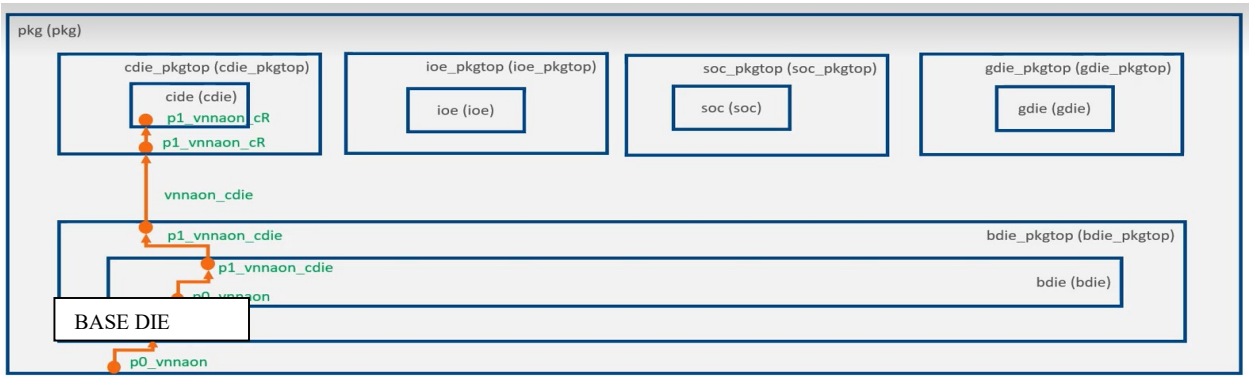

pkg (pkg)
cdie_pkgtop (cdie_pkgtop)
cide (cdie)
p1_vnnaon_cR
p1_vnnaon_cR
ioe_pkgtop (ioe_pkgtop)
ioe (ioe)
soc_pkgtop (soc_pkgtop)
soc (soc)
gdie_pkgtop (gdie_pkgtop)
gdie (gdie)
vnnaon_cdie
p1_vnnaon_cdie
bdie_pkgtop (bdie_pkgtop)
p1_vnnaon_cdie
bdie (bdie)
BASE DIE
p0_vnnaon

Figure 4

II. HELPFUL HINTS

D. Abbreviations and Acronyms

1. D2D – Die To Die
2. RTL – Register Transfer Logic
3. UPF – Unified Power Intent
4. Universal Chiplet Interconnet Express
5. CXL – Compute Express Link

IV. RESULTS

Disaggregated Dies Connectivity Validation

Handling of multiple dies in a single package needs advance package level validation and existing structural tool doesn't support die to die connectivity validation. There are essentially three components in a multi die SoC
standard chiplet interfaces system description and interconnection using high-level language primitives a solver for design space exploration and D2D interconnect optimization
Therefore, a new methodology was developed which could ensure the correctness of interconnects and it consists of following

- Die-to-Die (D2D) and Die-to-Package (D2P) connectivity
- User Defined ports names at Die interface (due to back port)
- Python Connectivity Files
  Checkers –LINT, Low Power, IFCDIFF, FC_SYN, SPA_COMPARE, Localize UPF, pkgball ports checker, top/ bottom side port checks, PROBE checks
- Reports - Tieoff, Pinlist.xlsx, Web Dashboard
- Outputs – RTL, UPF, Handoff to Backend collateral, Alias RTL for validation

Figure 5 shows an example of TieOffs that are caught during the package level connectivity validation and using the disaggregation tool, it can be identified at the front end level itself.

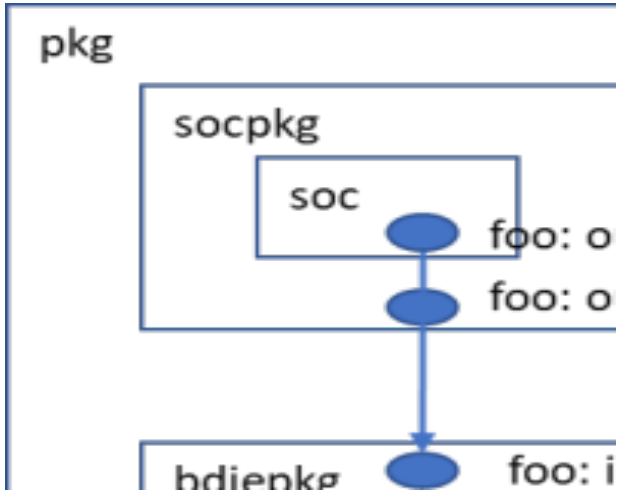


Figure 5

CONCLUSION

Robust Die-to-die (D2D) interconnect is key to having a successful solution for disaggregated strategy. Numerous factors need to be considered to arrive at most optimal chiplet interconnection that meet product landing zones and provide the right level of abstraction with EoU to create and manage product configurations built using mix-n-match of chiplets. Tool is first of its kind novel solution to automate interconnect creation and optimization. Tool provides robust mode-of-work for asynchronous DIE design across project milestones.

The execution of the first generation of server disaggregation project has been full of challenges concerning FDI design deliverable schedule and quality due to the IP widespread across multiple chipletsThe new methodology resulted in the below advantages over traditional excel based connectivity validation

- Utilization of standard interfaces and high-level primitives for chiplet interconnect creation (IMPACT: 6X ↓ in # of lines of code)
- Provision of robust automation and a dashboard (RESULT: 3X ↓ # of Iterations between Front-End & Back-End Teams)
- Enablement of easy-to-create reusable interfaces for creating product configurations (IMPACT: 4X ↓ net-net

engineering effort)
• Provision of an end-to-end methodology for cross-disciplinary teams to optimize their workflow

REFERENCES

[1] https://www.anandtech.com/show/17288/universal-chiplet-interconnect-express-ucie-announced-setting-standards-for-the-chiplet-ecosystem
[2] https://defactotech.com/
[3] https://www.uciexpress.org/
[4] https://semiengineering.com/standardizing-chiplet-interconnects/