Concurrent Planning and Feasibility for Efficient Package-on-Package (PoP) Design

Kevin Rinebold – Product Manager
Sigrity, Inc.
rinebold@sigrity.com
(408) 260-9344

Introduction

3D semiconductor packages enable the form and function for many of the devices we use in our daily lives - mobile handsets, personal entertainment devices, and flash drives to name a few. For some, these 3D packages play a critical role in improving quality of life as they rely on implantable medical devices like insulin pumps or defibrillators. Functional density, weight, and configurability are just a few of the reasons why a growing number of semiconductor products are going vertical using stacked-die, Package-on-Package (PoP), or Thru-Silicon-Via (TSV) packages. However, targeted design planning, implementation, and analysis strategies are required to achieve the full potential of these approaches.

PoP is one of the fastest growing packaging formats with a 40% CAGR projected through 2012 according to TechSearch International. [1] Ability to test at the package level and ease of multi-sourcing make it a popular choice with OEMs but it necessitates careful coordination and design planning. A typical PoP may contain a baseband device or media processor in the bottom package with some form of memory in the top package *(Figure 1)*. It's likely the memory is a standard catalog device with fixed pin assignments per the JEDEC JC63 standard so there's minimal flexibility with its package layout.

The interface or landing pads located on the top surface of the bottom package are the contact points between the two packages. Factors like IO pad ring layout of the base chip, differences in net names, die attach method, routeability of the bottom package, and host printed circuit board factor greatly into the package-package interface. Most design flows attempt to address these factors during physical implementation of the bottom package but given the multi-substrate nature of the problem it's too late and often results in numerous iterations and less than optimal designs. This is attributed to traditional methodologies developed for sequential design flows that use separate tools and databases for the packages and chips.

The key to efficient PoP devices starts with design planning when the options to effect change are greatest and cheapest to implement. Finding and fixing problems while still in the design planning stage avoids overdesign and its associated cost impact. PoP planning should take place prior to, or concurrent with chip floorplanning since the IO pad ring layout of the base chip and the package-package interface pads are directly related. Ideally the package-package interface becomes the starting point for design planning letting the memory devices dictate pad placement then modifying IO pad ring placement as necessary. This type of concurrent design planning requires simultaneous access to design data for the base chip as well as both packages.

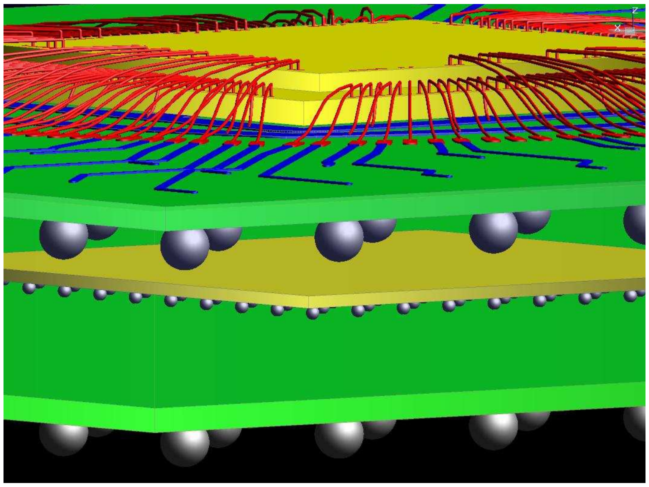


Figure 1 - PoP with stacked-die in top package and flip-chip device in bottom package

Tool and Methodology Limitations

Attempting any type of coordinated design planning across the chip, package, and board using traditional tools and serial methodologies can be challenging and frustrating at best. One problem is separate design environments and databases – one for the chip, a second one for the package, and a third for the board. [2] Even in this situation it's not uncommon for design teams to collaborate using spreadsheets to communicate pin assignments. The short-coming is it's based on snap-shots of static data, resulting in a highly iterative, error-prone process that does little to reduce cycle-time or cost of results

Package level integration in the form of PoP (or SiP for that matter) creates additional challenges for traditional tools and methodologies. [3] The multi-chip aspect of these packages adds the dynamic of chip-to-chip connectivity in addition to chip-to-package. Designers often use the fixed IO of one chip to influence IO and connectivity assignments on adjacent chips. While chip floorplanning and implementation tools work well for their intended application, they lack the ability to deal with multiple chips simultaneously. On the other hand, package- and board-level tools that support multiple chips lack the needed gate and macro visibility necessary for chip-level IO pad ring placement and assignment.

New Solutions for Design Planning and Feasibility

A new generation of IO planning solutions, such as Sigrity's OrbitIO Planner, takes a revolutionary approach, bringing all data sources together into a common, unified planning environment. Placement and connectivity scenarios are easily derived and evaluated in the context of the full system. Feasibility functions provide the means to incorporate aspects of detailed implementation while still in the early stages of design planning. A unified chip-package-board data model facilitates the seamless flow of data between domains allowing changes to automatically propagate to adjacent designs where their impact can be immediately evaluated. This ability to optimize the IO and connectivity design plan for performance, cost, and manufacturability prior to detailed implementation will result in fewer and faster design iterations with an overall reduction in complexity and cost.

IO planning and feasibility tools for PoP require innovative functionality to manage and manipulate a range of data at various stages of completeness. Ease of implementation and usability are crucial because these tools must plug into existing flows with minimal disruption. They must be able to instantiate design information on-the-fly for timely planning and feasibility in the absence of detailed data. Solutions must be vertically-aware to support PoP and other 3D package formats as well as provide the versatility to model die attachment scenarios utilizing wire bonding or flip-chip. In the case of OrbitIO Planner there are four major components that comprise the solution; data management and integration, device placement, feasibility tools, and connection planning.

Data Management

The ability to make placement and net changes in one domain and immediately see the impact on adjacent domains is made possible by an underlying data model that unites chip, package, and board data. The unified data model not only serves as a data repository, but also tracks the originating data formats and any incremental changes to ensure proper back-annotation.

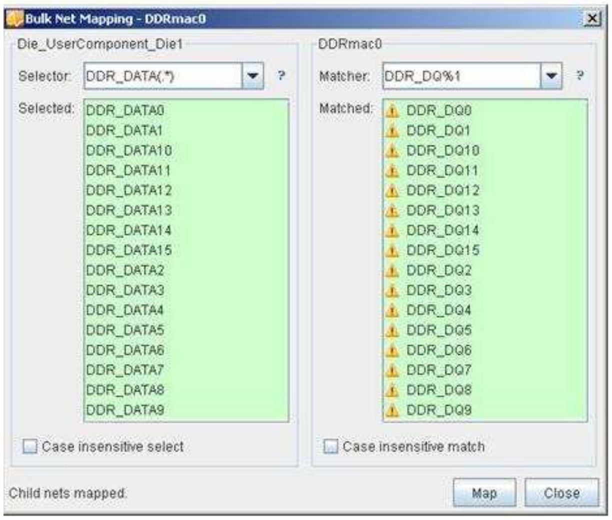
Populating the unified data model is accomplished utilizing a variety of standard data formats. File formats such as LEF/DEF, Verilog, and VHDL are the common data sources related to the base chip. LEF/DEF readers accommodate various data constructs and are capable of extracting the pertinent IO related information to avoid excessive data size. Verilog/VHDL readers possess intelligence to identify discrepancies between existing physical data and incoming logic. Spreadsheet support for IO pad ring definition is still necessary as companies take incremental steps to improve methodology. Native file formats from Sigrity, Cadence, or Zuken are sometimes used for packaging data. In their absences, an industry standard format like AIF can be used or even a simple spreadsheet that describes the ball pad map of the BGA.

During feasibility studies or early design planning, it's not uncommon for design decisions to be made in the absence of IO libraries or detailed IP information. Advanced IO planning tools provide capability to instantiate missing or virtual data on-the-fly to work through missing or incomplete design content. This can range from defining a simple die outline or BGA pattern, to adding IO pads cells on-the-fly, or even defining a placeholder for a chip-level macro. As detailed content becomes available the virtual content is replaced and validated using the tool's engineering change functions.

Functionality to process engineering changes is utilized as design content is constantly changing during design planning. These functions are capable of processing incremental updates so previous work is preserved. Common change scenarios include; LEF template updates, deleting or replacing contents of a device, net list changes, replacing one device with another, or the simple deletion of a device. In certain situations a compare and merge function can be helpful in identifying data differences.

Once the unified data model is populated, the relationships between devices must be managed, i.e., which chips go in which package and in what order. Automated hierarchy management is the fabric that ties everything together in the IO planning solution. It enables representation of the complete system from the gate level through the PCB while maintaining integrity of the individual designs. Almost every function within the IO planning tool will references the hierarchy before performing its designated task

When designs from different sources come together it's not unusual to encounter differences in net name conventions. It's not uncommon for a logical net to be referred to by different substrate specific net names. These differences must be mapped and correlated between domains before IO panning can take place. This is done through automated pattern recognition or with guidance from user specified regular expressions *(Figure 2)*.



Selector: DDR_DATA(.*)

Matcher: DDR_DQ%1

Selected:	Matched:
DDR_DATA0	⚠ DDR_DQ0
DDR_DATA1	⚠ DDR_DQ1
DDR_DATA10	⚠ DDR_DQ10
DDR_DATA11	⚠ DDR_DQ11
DDR_DATA12	⚠ DDR_DQ12
DDR_DATA13	⚠ DDR_DQ13
DDR_DATA14	⚠ DDR_DQ14
DDR_DATA15	⚠ DDR_DQ15
DDR_DATA2	⚠ DDR_DQ2
DDR_DATA3	⚠ DDR_DQ3
DDR_DATA4	⚠ DDR_DQ4
DDR_DATA5	⚠ DDR_DQ5
DDR_DATA6	⚠ DDR_DQ6
DDR_DATA7	⚠ DDR_DQ7
DDR_DATA8	⚠ DDR_DQ8
DDR_DATA9	⚠ DDR_DQ9

Child nets mapped.

Figure 2 – Mapping net names between designs

Tracking incremental changes as well as remembering original data formats is a function of the unified data model that comes into play when it's time to export data. The target tool and scenario will dictate the scope of content and file format. Some scenarios may require incremental changes to be exported as a script for an IC layout tool while others may require complete content in LEF/DEF or spreadsheet formats. There are situations that require filtering of content as in the case of excluding physical-only or PG devices from a Verilog file. Exporting die pad content from the IO pad ring requires collapsing multiple device instances into a single multi-pin component for use in package implementation tools. Not only do export functions support a variety of formats, they possess intelligence to recognize hierarchy and present the appropriate options.

Device Placement

A combination of automated and interactive features is used for device placement and optimization. These features are transparent across domains and easily applied to various situations. Adaptability to tasks ranging from rough IO pad ring layout to detailed placement of overlay cells is required. Results adhere to substrate-specific technology rules or region-based personalities, as in the case of multiple voltage planes. Device placement and pin assignments consider requirements for high-speed interfaces that utilize differential signals or special net topologies.

Pad ring placement can be accomplished in a variety of manners and is dictated by design flow and methodology. In some flows, initial placement originates from a spreadsheet while others instantiate and place IO pad cells on-the-fly to construct the pad ring. IO planning tools include functionality to automatically place IO pad cells for specific scenarios. For example, the ability to tightly pack cells together to determine minimum die size or ability to optimize cell spacing for wirebond technology

Sequence based automated placement helps expedite pad ring creation for flows that instantiate and place IO pad cells on-the-fly. A user-definable sequence file contains multiple strategies or recipes defining how various cell types are placed relative to one another. Sequences range from simple ordering of signal, power, and filler cells to more complex sequences that define ordering for a DDR2 interface.

Feasibility

Effective IO planning for PoP requires insight into aspects of the detailed layout while early enough in the planning stages to effect meaningful change. Wirebond configurations or flip-chip escape routing behave as intermediate connection points between die and package pads. These points can act as a redistribution mechanism with the unintended consequence of reordering the connection schedule. Advanced IO planning tools include innovative functionality to properly model these scenarios to ensure realistic and usable results.

Modeling wirebond configurations not only impacts connection planning, but also helps validate the quality of a pad ring layout. Converging on a mutually acceptable pad ring layout can be a major time-sync and is one of the leading causes of overly complex package layouts. Evaluating wirebond manufacturability while still in the pad ring layout stage will lead to optimal configurations in a much shorter timeframe.

If the chip in the bottom package utilizes flip-chip attachment the requirements for redistribution layer (RDL) and bump escape routing must be considered. RDL routing takes place on the upper most layers of the chip and is used to connect the IO pad cells to their respective bump cells *(Figure3)*. The required route resources are a direct result of the placement quality of the pad and bump cells. Bump escape routing is preformed on the package substrate and directly impacts layer count and design complexity. Via placement for the escape routes not only impacts routeability, but greatly influences connection scheduling. Just as the case with wirebonding, evaluating flip-chip feasibility in conjunction with IO planning will lead to shorter cycle-times and optimized designs.

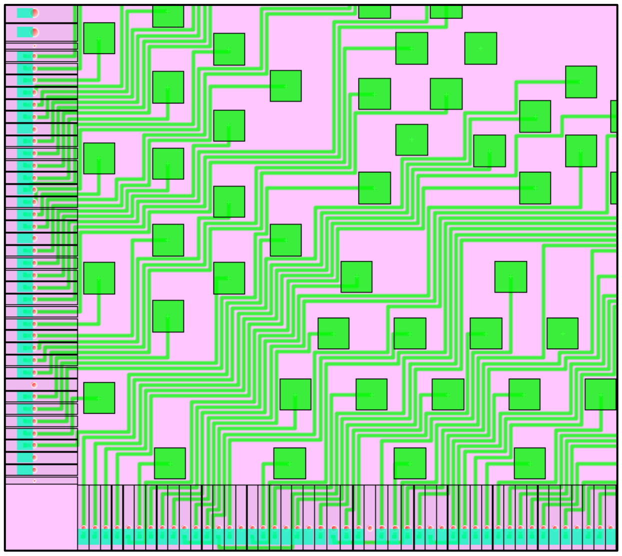


Figure 3 – Redistribution layer routing (RDL)

Connection Planning

Connectivity planning and creation functions work in concert with the placement and feasibility features of the IO planning tool. These are not isolated events but rather a tightly intertwined puzzle with the results of one impacting the others. Deriving balanced solutions requires a great deal of automation and is scalable to various situations across interconnect domains.

Consider the example of propagating connectivity from flip-chip bumps to package ball pads in the bottom package. One objective is achieving the shortest connections with the least number of cross-overs. Use of differential signals will mean some nets must be on adjacent ball pads to minimize skew. In turn, this can trigger a need for PG balls to be adjacent to the diff pairs. PG ball pad placement must consider the location of its respective voltage plane. This quickly becomes a multi-faceted problem that uses innovative functions to evolve a solution.

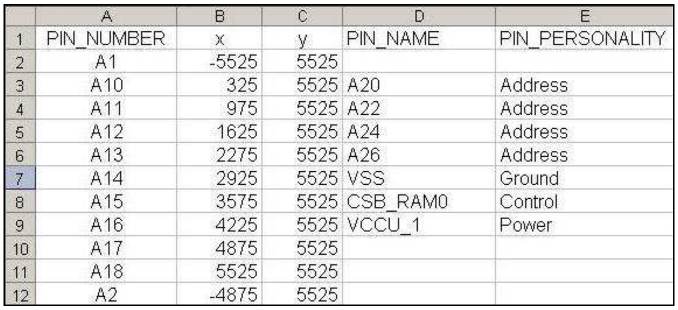
Advanced IO planning tools include robust mechanisms to properly describe the complex interactions and behaviors of connection planning. This starts with the automated definition of differential signals or defining the ratio of signal/power/ground pads. For flip-chip, connections must comprehend their routing layers or specific route topology. Mechanisms are used to group design objects based on common characteristic independent of net assignments. For example, the ability to reserve specific pin groups for certain types of signals.

Application of Concurrent Planning on a PoP Design

A common issue with PoP is coordinating the memory interface from the top package to the chip in the bottom package. Pin assignments on the top package have minimal flexibility since they typically adhere to the JEDEC JC63 standard for memory packages. The goal is to arrange the IO pad cells on the bottom chip in a manner resulting in the most direct connection path between devices. What makes this seemingly simple task so challenging is the lack of EDA tools that can coordinate database content between the multiple packages and die that comprise a PoP device.

As presented in this paper there is a new generation of EDA tools like Sigrity's OrbitIO Planner targeting these types of design planning problems. Disparate design data is brought together in a unified environment for coordinated planning and evaluation of design feasibility. Once the planning and feasibility steps are complete, updated design content is pushed back to their respective design tools for physical implementation.

The first step in the process is importing design content into the tool. In this example design content for the top package is defined as a simple spreadsheet consisting of pin names, XY locations, and net names (*Figure 4*). No information or description of the top die is included nor is it necessary since the package pin-outs are fixed. Design content of the bottom package is defined in an AIF file that was written from Cadence APD. Both the spreadsheet and AIF are easily imported in their native formats.

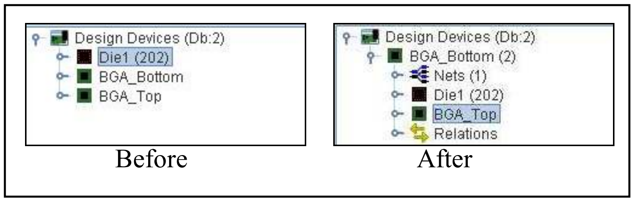


	A	B	C	D	E
1	PIN_NUMBER	x	y	PIN_NAME	PIN_PERSONALITY
2	A1	-5525	5525		
3	A10	325	5525	A20	Address
4	A11	975	5525	A22	Address
5	A12	1625	5525	A24	Address
6	A13	2275	5525	A26	Address
7	A14	2925	5525	VSS	Ground
8	A15	3575	5525	CSB_RAM0	Control
9	A16	4225	5525	VCCU_1	Power
10	A17	4875	5525		
11	A18	5525	5525		
12	A2	-4875	5525		

Figure 4 – Spreadsheet information of top memory package

The die for the bottom package is constructed using LEF files and a spreadsheet that defines the relative sequence of IO cells. The initial sequence of IO cells was defined by the ASIC designer based on core connectivity requirements. Some consideration was given to the memory interface to roughly locate the IO pad cells along the proper side of the die. Final location of the IO pad cells will be derived in the planning tool.

At this point the two packages and bottom die exist as three separate design databases in the planning tool. Establishing relationships between these designs is accomplished through a simple drag-and-drop operation in the device hierarchy manager (*Figure 5*).



Design Devices (Db:2)
- Die1 (202)
- BGA_Bottom
- BGA_Top

Before

Design Devices (Db:2)
- BGA_Bottom (2)
  - Nets (1)
  - Die1 (202)
  - BGA_Top
  - Relations

After

Figure 5 - Establishing relationships between devices

The next step addresses differences in net name syntax between the top package and the bottom die. In the case of the top package the nets for the address bus use a syntax of A[0-26] with the data bus using D[0-15]. The same nets in the bottom die are defined as ADDR[0-26] and DATA[0-15] respectively. Net mapping functions of the planning tool are used to establish equivalent net names between these devices while maintaining individually of the respective net lists.

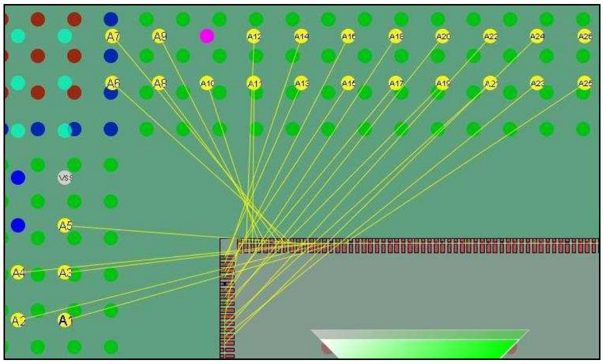


Figure 6 – Before IO pad cell optimization

With the nets properly mapped, connectivity for the memory interface is now visible (*Figure 6*). It shows the corresponding IO pad cells are in the general vicinity of where they need to reside but are in need of significant optimization to avoid package substrate routing issues. Optimization is preformed using the ball pads of the top package as the fixed anchor point and allowing the IO pad cell devices to swap locations within the die. The result is a more direct connection path requiring fewer route resources on the bottom package substrate (*Figure 7*). Once connection planning and feasibility is complete the package and die information is exported in domain specific formats.

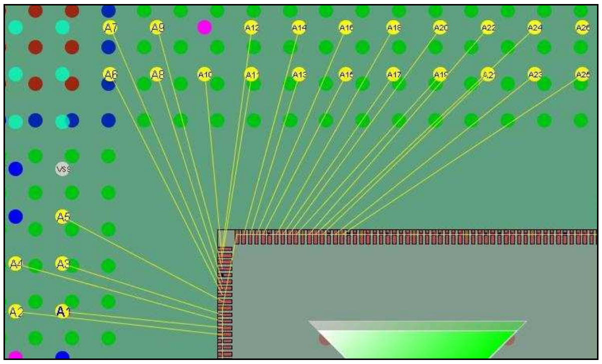


Figure 6 – After IO pad cell optimization

This example may seem simplistic but consider that both die and package data were manipulated within the same tool. Ability to quickly iterate over multiple data sets within one environment not only results in shorter cycle-times, but also contributes to reduced design complexity and cost.

Conclusion

Volumetric packaging like PoP continues to deliver the functional density, weight, and configurability needed to keep pace with Moore's Law for the foreseeable future. On the horizon, TSV packages promise even greater density and performance; however, lack of EDA tools is a limiting factor to their widespread adoption.

Progressive EDA vendors recognize the multi-substrate, multi-domain nature of 3D packaging and view the status-quo of sequential design flows with separate tools and databases as a severely limiting factor. The new generation of planning and feasibility tools supports critical decision making on issues that impact performance, complexity, and cost at a point in the design process when it's most economical to effect change.

References

1. TechSearch International Inc., "Advanced Packaging Update: Market and Technology Trends," March 2008.
2. Durrill, S. and Felton, K. "Methodology and Flow Challenges in Multi-die Package Design," *Advanced Packaging* , January 2007.
3. Smith, L., "Challenging Applications Driving Package Innovations – Case Study of Solution in Emerging Technologies," *Proc. MEPTEC Packaging Development and Innovations Conf*, November 2008.