Chip/Package/Board Interface Pathway Design and Optimization

Tom Whipple
Product Engineering Architect
November 2015
Chip/package/board interface pathway design and optimization

PCB design with Allegro® PCB layout environment

Package design with Cadence® SiP Layout packaging

IC design with Innovus™ Implementation System or Virtuoso® custom design platform
Interface pathway design and optimization in OrbitIO system planner
Key flows

- I/O assignment and optimization
- Variant design
- System topology exploration
- Design revision
- Integration with layout tools
Why pathway design and optimization?

- Design in the context of chip, package, and board
- Reduce layer count of board and/or package
- Reduce overall turnaround time
- Improve signal and power integrity
- Reduce design margins
Key Elements for Pathway Design
Design environment

- Core functionality with two design environments
  - OrbitIO™ design environment for hierarchical system-level exploration, pin creation, and optimization
  - Cadence® SiP Layout package layout tool for design layout and refinement
Hierarchical database

- Preserve technology, layer stack-up, and net names for each device in the hierarchy
- As signals traverse devices, net names may change
- Provide net graphing to easily see net throughout whole system
Start from nothing, start from existing

• Most layout tools require full layer stackups, full netlists, complex constraint setups
  – Provides a formidable barrier for early pin creation and optimization
  – Forces a serialization of the design flow

• OrbitIO™ system planner supports:
  – Start-from-nothing flows
    – Create components as you go
  – Start-from-existing flows
    – Import parts from existing libraries (.dra, .def, die abstract, .csv)

• Allows for a more parallel design process
Ball, bump, and I/O pad ring design

- Create or import a wide variety of standard or custom pin patterns and I/O pad rings

- Support LEF/DEF, .dra, .csv, die abstract, die/bga txt

- Create patterns of signal, power, and ground to guide net assignment
Routing pathway definition with bundles

- How do you optimize the nets between point A and point B, in a PCB or in a package?
  - Manually route each line from point A to close to point B, then swap pins to minimize rat’s nests crossings?
  - Temporarily move components close to each other, optimize the pins, then restore the original placement?

- Cadence recommendation:
  - Bundle signals by function (DDR, SerDes, USB, etc.) or by proximity
  - Use a bundle to define routing topology from A to B
    - The bundle preserves signal order throughout pathway
  - Breakout pins from both points A and B
    - Use any existing breakout routing
    - Optimize pin assignments

- A hierarchical environment allows optimization of nets from PCB components up through to die bumps and drivers
Routing pathway definition with bundles example
Constraints

- Must keep it simple!
- Early stage doesn’t deal with full-scale routing, so no need for full-scale constraints
- How much is enough?
- Interfaces—keep signals grouped according to function
- Width, spacing, differential pairs
- Interface definitions and diff pairs must transfer between tools
Schematic integration for PCB designs

- When changing symbols in a physical layout that originates from a schematic, the schematic must also be kept in sync

- Flow:
  - Create or update schematic symbols
  - Re-package from schematic to layout
  - Import bundles from OrbitIO™ system planner

- Allegro® Design Entry-HDL is supported today, OrCAD® support is in development

- For early exploration, write directly to layout
  - Bypass schematic overhead at this stage of the design
Interoperability

• The hierarchical pathway design environment and the physical layout environments must efficiently talk with each other!

• Binary exchange between tools is standard practice
  – Allegro® PCB and Cadence® SiP Layout read OrbitIO™ database directly
  – OrbitIO system planner reads .mcm, .pcb, and .sip files directly
  – From a vendor standpoint, this is very efficient

• Die abstract is used to exchange die information between Innovus® and Virtuoso® design environments and system design and layout tools.
Pathway Design Flows
General flow

• Import existing designs or libraries if they exist

• Create design elements that aren’t yet available

• Define design hierarchy

• Define net mapping between parts

• Define interfaces and differential pairs

• Optimize from fixed pin configurations to ball or bump arrays

• Use route feasibility to validate component breakout can be done

• Export parts to layout environment for device implementation & analysis
Pin optimization

• This is a key pathway design feature

• Start from known, fixed pin configurations
  – Board component breakout endpoints
  – Known die IP blocks
  – Existing BGA footprints

• Optimize the placement of I/O drivers in a die based on route topology from a component on the PCB
  – If you have defined component breakouts on a PCB component, the ordering of the signals to the breakouts can drive signal sequence throughout the design

• Generate spreadsheet ball map for documentation and design review
Exploring system topology options

- Explore tradeoffs for various configurations

- Determine if a part should be on the board, in the package, or on an interposer

- Drag and drop components from one level of hierarchy to another
Design revision (start from existing)

- Import previous design
- Add interfaces
- Change the footprint or package
- Re-optimize
- Easily explore options for new enhancements to the design
Exploring variants

- Explore placing a die in several package variations
- Explore placing several die variations in a given package
- Explore placing a package in several board configurations
Analysis—Two approaches

• Approach one:
  – Create models for each part (one for board, one for pkg, one for IC)
  – Merge the two together in the analysis tools for system-level analysis
  – This is how Allegro® and Sigrity™ systems work together
  – Simpler use model
  – Does not capture inter-component interference

• Approach two:
  – Combine all parts into one and create model for the whole system
  – Requires the physical layout environment to combine all substrates
  – More complex use model
  – Captures inter-component interference
Conclusions

• Pathway design environment allows non-layout experts to design and optimize package and die pins

• Planning of breakouts further optimizes routability

• Design-from-nothing and design-from-existing allows to begin pathway design early in the design flow

• Variant design flows provides greater visibility into all variants at once

• Reduce design costs by lowering layer count, shrinking design time, and providing greater predictability of meeting constraints in a broader design context