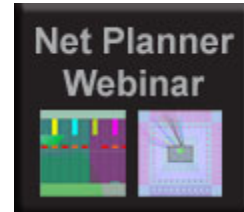


## A Low-Cost Task Specific Solution for IO Pad-Ring and Package Net List Construction

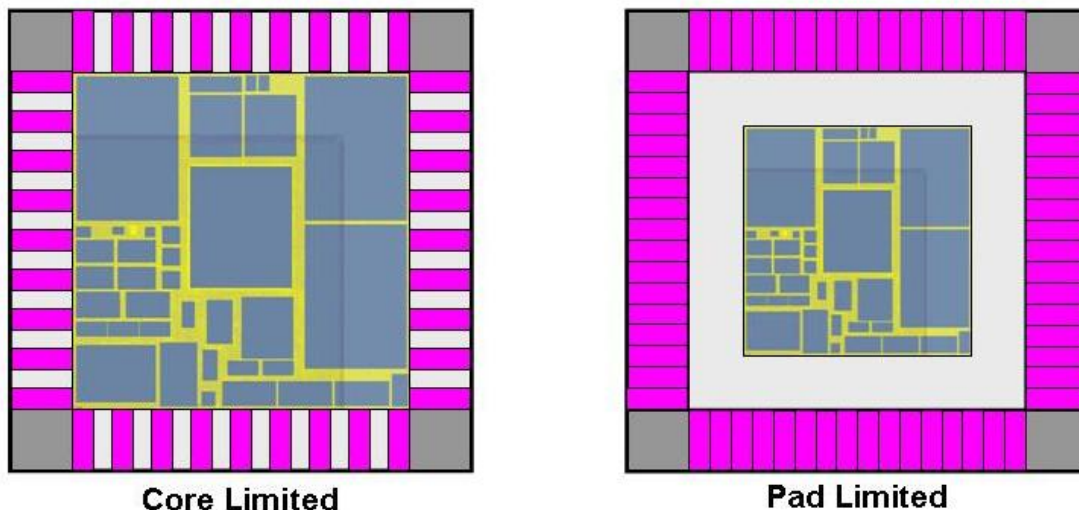
[Kevin Rinebold](#); Sigrity Senior Product Manager; May 2009

This paper will discuss the IO pad-ring and package net list construction process and the corresponding design methodologies companies typically employ. It explores the efficient application of resources and identifies key tool and design flow considerations. A new low-cost, task specific solution for pad-ring construction and die-to-package net list planning and optimization will be introduced. The tool provides visualization in an environment that promotes rapid development and exploration of the full solution space by incorporating package net list development concurrent with pad-ring construction.



### Managing Pad-Limited Design

Ideally die size is a function of the area consumed by standard cells and macros resulting in what's known as a core-limited design. These designs have sufficient space around the periphery to accommodate the required IO pad cells. However, as process geometries get smaller, more functional content can be designed into the chip resulting in a greater demand for IO. Due to this demand more designs are pad-limited where IO pad cell area is the determining factor on die size and not the functional core area (*Figure 1*). IO pad-ring development for a pad-limited design can be a highly iterative time consuming process in the search for the ideal die size and pad-ring configuration.



*Figure 1 – Comparison of core-limited and pad-limited designs*

While the IO pad-ring is the conduit to the world outside the chip, its development is often tangent to that of core functionality and as such receives minimal automation and tool support from major EDA vendors. The IO pad-ring placement is poorly addressed by standard floorplanning automation and almost never accounts for connections outside the chip. To fill this void companies develop their own solutions for IO pad-ring and package net list development. Methods vary but they share the common trait of using spreadsheets, scripts, or proprietary tools.

A typical flow will use Excel to formulate the pad-ring in spreadsheet form which is then imported into the IC implementation tool via custom script. An alternate approach is use of an internally developed tool in place of Excel to construct the pad-ring spreadsheet. The content and sophistication of pad-ring spreadsheets will vary but in essence it's used as a non-graphical placement vehicle. Visual feedback on the resulting placement doesn't occur until the spreadsheet is imported into the IC implementation tool. In some companies one Engineer is responsible for both pad-ring formulation and running the IC tools while at other companies these functions cross departmental boundaries or areas of responsibility.

Pad-ring content and structure combined with Excel's availability and ease-of-use makes it natural choice for editing. Features like cut-and-paste, auto-increment, and ease of moving data blocks can help expedite pad-ring construction. However the lack of graphical feedback can slow development forcing users into IC tools for simple visualization. While this approach may work there are numerous factors and drawbacks to consider.

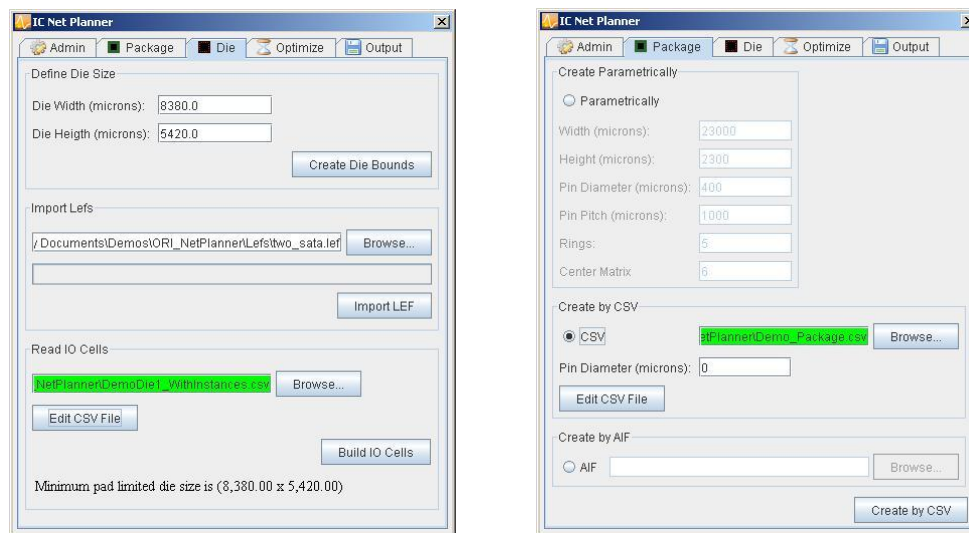
- How many iterations are triggered by spreadsheet typos or the need see what's happening? What's the cycle-time impact?
- Do using IC implementation tools for pad-ring construction represent best use of high-value software assets?
- Can the IC tools support a high level of abstraction or do they require well formed inputs? What happens if IO libraries are missing or interfaces are loosely defined?
- What's the cost and support overhead of internally developed tools?
- Does the flow support rapid prototyping of die size and pad-ring configurations in response to feasibility studies?
- How is the package incorporated into the pad-ring construction process? What about management of differential signals? How well do the chip and package design teams collaborate?

### **About OrbitIO IC Net Planner**

OrbitIO IC Net Planner is a low-cost, task specific tool for pad-ring construction and die-to-package net list planning and optimization. Its core technology is derived from Sigrity's OrbitIO product family of codesign tools. These tools reflect a revolutionary approach to codesign that combines design feasibility engines with a complete view of chip-package-board connectivity all within a unified codesign environment.

IC Net Planner builds on the concept of spreadsheet based pad-ring construction adding an interactive graphical environment that promotes rapid development and exploration. It incorporates the physical definition of the package which allows die-to-package net list development to take place concurrent with pad-ring construction. IC Net Planner takes the pad-ring from a high-level of abstraction and distills it down to detailed content ready for use in implementation tools.

IC Net Planner employs a single multi-tab dialog that exposes functions upon satisfying prerequisite steps to help guide users through the construction process (*Figure 2*). The functional tabs are reentrant allowing changes at any point that in turn automatically update the design, thus easing the burden of an ECO cycle. Input files are time stamped and tracked to ensure use of the latest data. If subsequent edits occur, the affected files are flagged in the user interface.



**Figure 2 – OrbitIO IC Net Planner’s dialog for die and package construction**

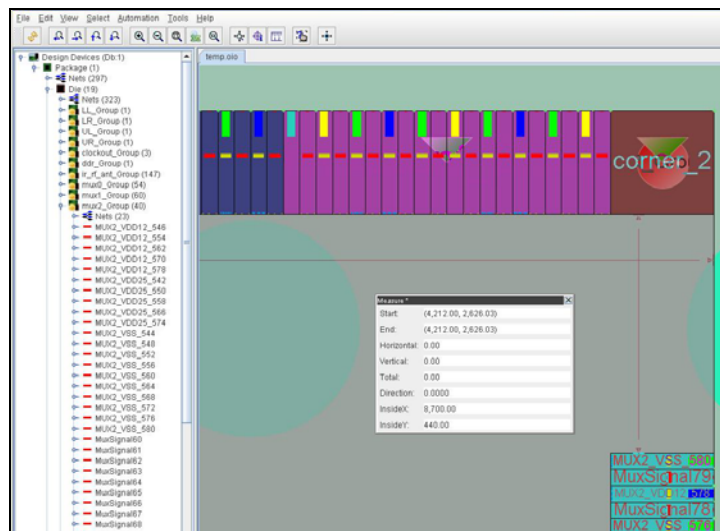
IC Net Planner is comprised of five key components; die and pad-ring construction, package construction, net list generation and optimization, interactive review and refinement, and domain specific outputs.

- *Die and pad-ring construction* – a spreadsheet containing pad-ring information is used in conjunction with LEF files to assemble the pad-ring. IC Net Planner is highly flexible in the structure and content of the spreadsheet, and provides an integrated launch of the spreadsheet editing program. Typical content includes device instances, templates, die side, and net information. Additional content may include transform controls, pin and device grouping, support for multi-pin devices, and differential signals (*Figure 3*). Placeholders are easily inserted in the case of missing IO libraries or loosely defined interfaces. Die size is defined interactively and can be modified at any time.

A	B	C	D	E	F	G	H	I	J	K	L	M
Personality	CellInstanceName	NetName	TemplateName	Side	PinName	PinType	PinPersonalityName	DiffPair	Width	Height	Gap	Post
mux3	MuxSignal117	MuxSignal117	PDB08DGZ	top	PAD	WIREBONDPAD	SIGNAL					
mux3	MUX3_VSS_857	MUX3_VSS	PVSS2DGZ	top	PAD	WIREBONDPAD	VSS					
refclock	CLKE1	CLKE1	PDB08DGZ	top	PAD	WIREBONDPAD	SIGNAL					
refclock	VDD12_867	VDD12	PVDD1DGZ	top	PAD	WIREBONDPAD	VDD12					
refclock	CLKT1	CLKT1	PDB08DGZ	top	PAD	WIREBONDPAD	SIGNAL					
refclock	VSS_869	VSS	PVSS1DGZ	top	PAD	WIREBONDPAD	VSS					
usb2	PVDDIO_0	PVDDIO	PVDDIO	top	PAD	WIREBONDPAD	VDD12					N
usb2	\$SCB_CLKIN0	SCBCLKIN0	LVDSRCVR	top	PADN	WIREBONDPAD	SIGNAL	dp1				FN
usb2	\$SCB_CLKIN0	SCBCLKINP0	LVDSRCVR	top	PAD	WIREBONDPAD	SIGNAL	dp1				FN
usb2	PVSSIO_0	PVSSIO	PVSSIO	top	PAD	WIREBONDPAD	VSS					N
usb2	\$SCB_DATAIN0	SCBDATAIN0	LVDSRCVR	top	PADN	WIREBONDPAD	SIGNAL	dp2				FN
usb2	\$SCB_DATAIN0	SCBDATAINP0	LVDSRCVR	top	PAD	WIREBONDPAD	SIGNAL	dp2				FN

**Figure 3- Example of a pad-ring spreadsheet**

- **Package construction** – there are three options to define the package; parametric construction, import from a spreadsheet, or AIF. These options represent the most common methods of extracting package data from an external CAD System.
- **Net list generation and optimization** – connections are automatically created from the pad-ring to package ball pads based on several objectives. These include; matching pin group names between die and package, minimizing connection crossings, minimizing overall net length, reducing the angle of connection emanating from the die, placing diff-pairs on adjacent ball pads, and minimizing the difference in length between diff-pairs.
- **Interactive review and refinement** - an interactive canvas provides graphical feedback while a hierarchical tree-style display lists the contents of the pad-ring. Devices, nets, and pins are easily searched and located while inquiry and measurement utilities permit further review (Figure 4). Interactive pin swaps can be preformed to further refine the net list.



- **Domain specific outputs** – A DEF file of the die and pad-ring along with the LEFs of any instantiated devices can be exported for use in IC implementation tools. Package specific exports include; an AIF file for direct input into package design tools, a generic spreadsheet of ball locations and nets, and an HTML ball map. IC Net Planner also provides an HTML die-to-package net mapping table.

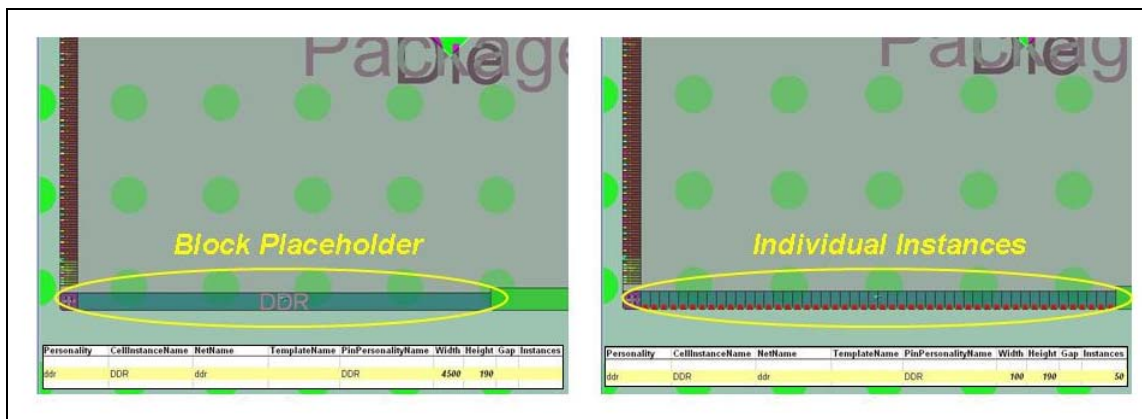
## Finding the Right Die Size and Pad-Ring Configuration

At smaller semiconductor process nodes it's not uncommon to encounter pad-limited designs where IO is the influencing factor on die size and not the core. Multiple combinations of die size and pad-ring configurations must be explored to arrive at the ideal balance of silicon utilization and manufacturability. How does one ring of IO pads compare to using multiple rings and what's the resulting impact on die size? IC Net Planner is the perfect environment to quickly iterate through these scenarios to develop a solution.

Die size is easily adjusted in IC Net Planner with the impact on pad-ring placement immediately displayed. It also reports the minimum pad-limited die size based on the spreadsheet definition. IO rings are adjusted via the spreadsheet and defined in relationship to their corresponding die side. For example, an entry of *Left1* in the die side column denotes the outermost ring on the left side of the die; *Left2* would denote a second ring and so on. The interactive measurement utility comes in handy when determining available space or distance between objects.

## Expedite Planning Even With Missing Information

An issue for tools requiring well-formed data is encountering loosely defined interfaces or missing IO libraries - which are common during early stages of development. In this situation IC Net Planner utilizes a placeholder defined via the spreadsheet that approximates size and area of the missing devices, helping expedite planning. As libraries or detailed content becomes available the design is updated to reflect the new information. In the following example a block is defined in the spreadsheet to reserve area for a DDR interface enabling continued development of the pad-ring (*Figure 5a*). When details of the DDR interface become clear, the block is then converted into individual instances ready for net list generation (*Figure 5b*).



**Figure 5a – Using a placeholder**

**Figure 5b – Individual cell instances**

## Request for Quotes and Feasibility Studies

Occasionally it's necessary to perform design feasibility studies in response to sales or marketing inquiries from a customer. This usually requires a quick pass at the die and

packaging configuration to estimate design effort and cost. In many cases information is sparse and arbitrary at best while others may use an existing design as a baseline for new features. IC Net Planner's ability to work with minimal information, quickly iterate die size and pad-ring configurations, and generate a package net list makes it an ideal solution for feasibility studies. Its also minimizes use of expensive software assets on a potential non-revenue exercise.

### **Jump Start Detailed Implementation**

The ability to take loosely defined data and transform it into well-formed design content is a key feature of OrbitIO IC Net Planner. Abstractions of the design can be shared at any point in the process by generating IC or package specific outputs. Snapshots of the pad-ring can be shared with IC design teams via DEF files while the package team can preview the net list via AIF file. Early dissemination of this information can help prevent downstream problems that otherwise go unnoticed until detailed implementation. The result is fewer iterations and shorter cycle-times.

Communicating differential signal assignments can be tedious task that slows package development. Automatic net list generation within IC Net Planner accounts for diff-pairs to ensure they're on adjacent balls with closely matched lengths. The resulting ball pad assignments are then communicated via AIF or generic spreadsheet format to the packaging team eliminating any ambiguity over assignments.

### **Smarter Use of Resources**

Most pad-ring development flows rely on IC design tools at some point in the process – some used prominently, others to a lesser degree. While these tools provide graphical feedback and insight into the die and pad-ring layout, they also represent some of the most expensive software assets used in the design process. The pad-ring will ultimately be consumed by these tools but they're not necessary for initial development. IC Net Planner represents a significantly lower-cost alternative for pad-ring development and in most cases delivers a greater level of functionality. It frees IC tool licenses to be applied to tasks that better justify their return-on-investment (ROI).

IC Net Planner is also an attractive alternative to internally developed tools for pad-ring development and in most cases represents a more cost-effective solution. The benefit of internal tools has been a targeted feature set that is well aligned with the company's design flow. With that comes the support the maintenance obligations to keep tools current resulting in costs that are increasingly difficult to justify as companies align on core competencies.

### **Conclusion**

Today's business climate is forcing companies to evaluate their processes to identify new ways to economize while remaining competitive. Just because a design flow works doesn't mean it's an efficient use of resources. In most scenarios there are hidden costs that manifest in the form of additional iterations, longer cycle-times, internal tool development, or additional IC design tool licenses.

OrbitIO IC Net Planner delivers a task specific solution for pad-ring construction and die-to-package net list planning in a low-cost product that can be incorporated into any flow while providing an attractive ROI. It builds on the familiar spreadsheet paradigm adding a powerful graphical environment that promotes rapid exploration of the solution space resulting in fewer iterations and shorter cycle-times. Domain specific outputs streamline data exchange to implementation tools helping expedite the design process.

### **More on the OrbitIO Product Family**

The objective of codesign or IO planning is to coordinate device placement with associated pin and net assignments within the chip-package-board system – prior to detailed chip implementation – when the options to effect change are greatest and cheapest to implement. It requires early insight into aspects of the detailed package layout and the ability to understand chip-level logic restrictions like hard macros or high-speed interfaces and their impact on IO pad ring layout.

The OrbitIO family of codesign and IO planning tools 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.