cādence®

Early Die Bump Planning using SiP Layout with EDIS

Product Version SPB16.6
December, 2015

Contents

Purpose

The purpose of this Application Note is to describe the process for early IC bump planning and data exchange based on DEF and die abstract using the SiP Layout and Encounter Digital Implementation System (EDIS) environments.

Audience

This application note is intended for Package designers and IC bump array designers responsible for the IC/package bump planning process. A method to intelligently define, manipulate and exchange the bump plan data between the SiP Layout and Encounter Digital Implementation System (EDIS) environment is demonstrated.

Overview

Given today's large Die bump counts, multiple power domains, small pin pitches and high speed interfaces, early die bump planning has become a critical part of the IC and package design process. In the past, the IC designers were typically responsible for the die bump process, and it was often done in the late stages of the IC physical design process due to IC designers' focus on RTL, timing, IP resourcing, IP/PHY/macro integration, and many other factors. However, there has been a shift in the recent years, and it is now a critical component that must be included in the early stages of the IC design process. To ensure a proper bump plan meeting the ultimate goals of the IC design requirements, and start the process of sizing Die, mocking up hard macros, io's, PHY, and core logic requirements, it is done even as RTL is being developed.

This is an extremely long and complicated process that requires the knowledge of IC and Package technologies. Currently, Pkg designers undertake this by using their packaging tools of choice to exchange data with the IC designers throughout the process.

The high-level key areas in the bump planning process should address the following minimum requirements:

- Consuming and managing the IC LEF information being used for the IC design
- Flexibility in designing and managing both UBM (flip chip passivation opening) and die pin requirements
- Integration of early IC top-level details (LEF technology) and Substrate materials
- Die Shrink planning and Pkg CTE bump pins for assembly planning
- Intelligent data exchange from the Pkg environment (SiP Layout) to IC (EDIS; Encounter Digital Implementation System)

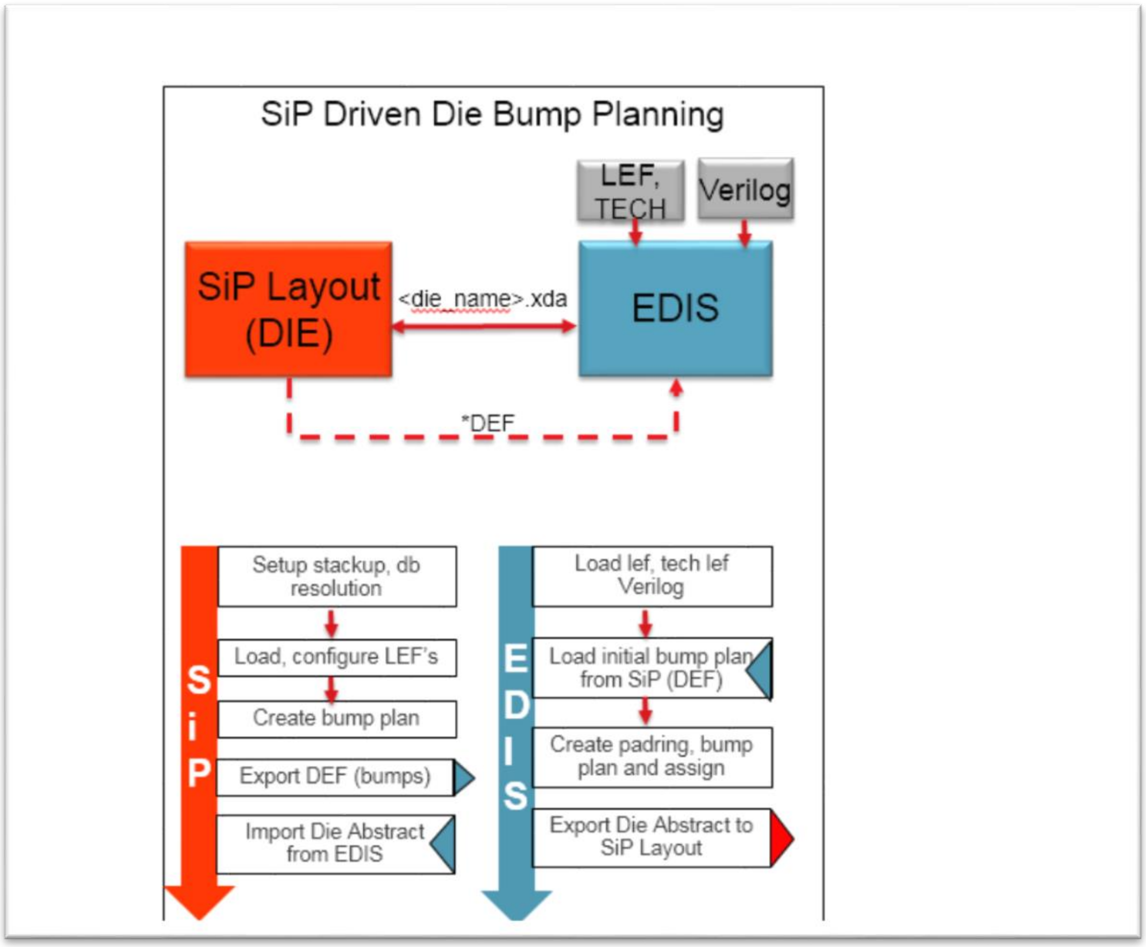
    - Must include ECO as die bump refinement process
    - Must leverage either IC netlist information or netlist generated on the fly at the Pkg level

Early Die bump planning using SiP Layout with EDIS

Before commencing with the actual bump plan flow process, it is important to understand the proposed top-level bump planning flow and introduce the die abstract data concept.

The following image shows the proposed flow solution covered in this document:

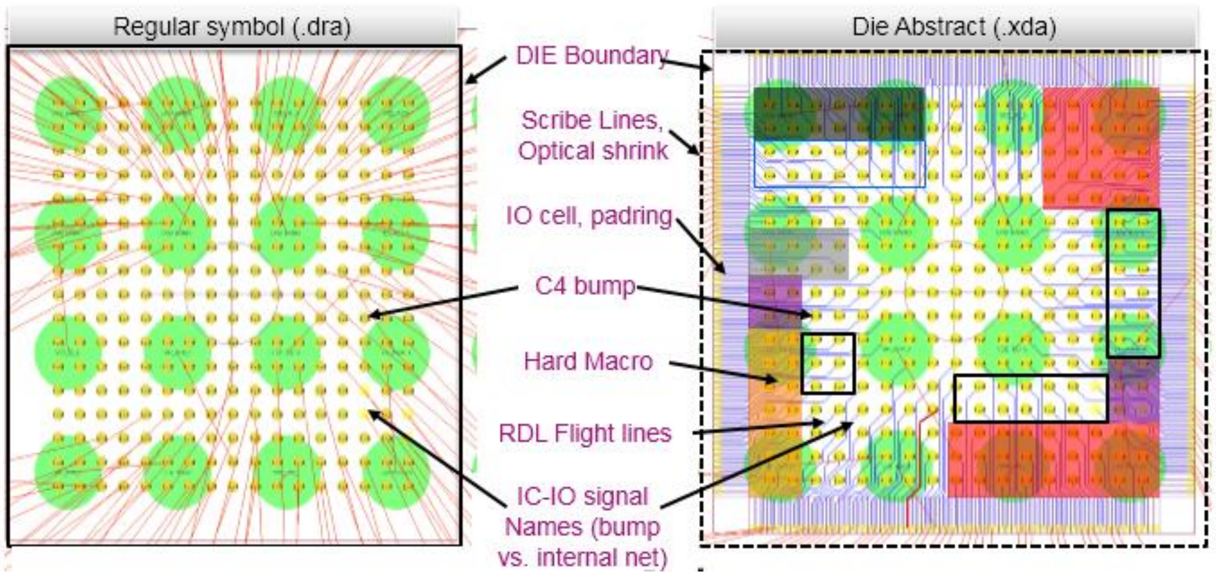
Flow Solution: Early SiP-driven bump planning


SiP Driven Die Bump Planning
LEF, TECH
Verilog
SiP Layout (DIE)
<die_name>.xda
EDIS
*DEF
SiP
Setup stackup, db resolution
Load, configure LEF's
Create bump plan
Export DEF (bumps)
Import Die Abstract from EDIS
EDIS
Load lef, tech lef Verilog
Load initial bump plan from SiP (DEF)
Create padring, bump plan and assign
Export Die Abstract to SiP Layout

In the flow, notice the key bump plan bidirectional data exchange process is based on the die_abstract.xda. The SiP Layout and EDIS environment have had this capability for many years. Furthermore, this flow assumes starting from a “ground up design flow”, meaning there is no IC die information available to the user. If an EDIS design is available with pre-planned bumps, the DEF process can be omitted, and the die abstract flow process is used as outlined in this application note.

A brief description and comparison with a simple SiP Layout footprint symbol instance model and that of the abstract.xda is described in the following image:

- Anatomy of a dra symbol vs. die abstract for co-design and co-analysis:
  - Dra symbol has no intelligence, only pins and footprint dimension, and text labels.
  - .xda embeds top-level IC data details for design, optimization and analysis.



Now that the baseline information has been described for both the bump plan flow and the abstract exchange mechanism, you can proceed to how the entire process works with the SiP Layout and EDIS.

Note: The SPB 16.6x release does not support the creation of a die abstract (abstract.xda) from a start from nothing flow. Only a die abstract initially generated from EDIS is supported as the bidirectional data exchange mechanism today. However, SiP Layout does provide the capabilities of generating the bump plan based on DEF. As such, we can employ the DEF bump plan pin out for a start from nothing flow, pass that initial DEF based bump plan to EDIS, and then use the abstract.xda process for the remainder bump planning process.

An alternative option is to start with a mockup bump array or a predefined bump array in EDIS, export EDIS die abstract, and load that abstract to the SiP Layout.

SiP Layout: Initial data setup using LEF

You can use the following options to set up the design information into the SiP Layout for early bump planning:

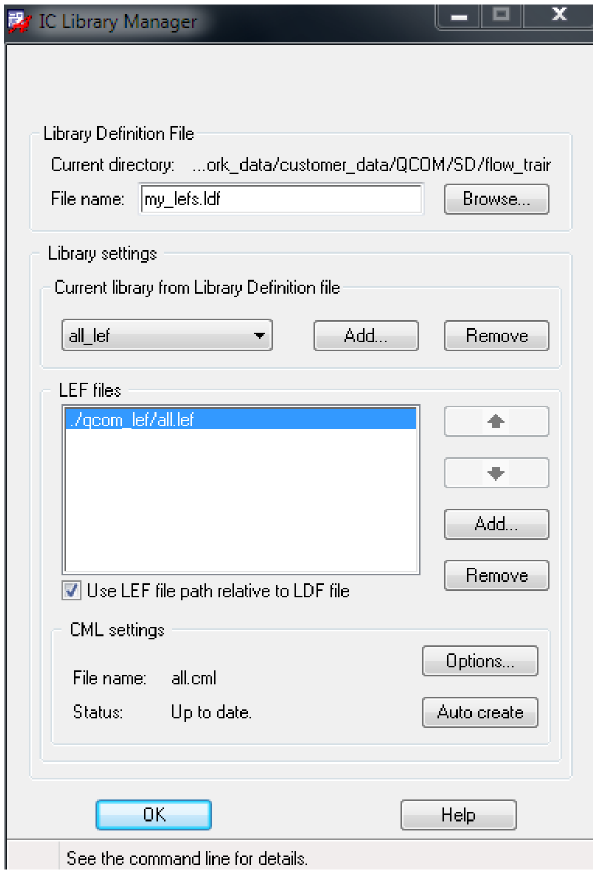
1. There are currently two options from available in SiP Layout to create a start from nothing flow for the initial bump plan and exporting that data to EDIS:

    a. LEF/DEF (Def out) based
    b. GDSII (stream_out)

2. You will use LEF/DEF out for this flow (only needed once, afterwards all based on die abstract):

    a. Load LEF file(s).
    b. Create/open your <design_name>.sip design.
    c. Select Setup→LEF Libraries.
    d. Add file name under library definition file.
    e. Add Library name under Library Settings
    f. Select Add and load LEF file(s).
    g. Select Auto Create under CML settings.
    h. Set the SiP Layout to Symbol Editor Application mode.

The following image shows the sample setup using the IC Library Manager:


IC Library Manager

Library Definition File

Current directory: ...ork_data/customer_data/QCOM/SD/flow_trair

File name: my_lefs.ldf

Library settings

Current library from Library Definition file: all_lef

LEF files: ./qcom_lef/all.lef

☑ Use LEF file path relative to LDF file

CML settings

File name: all.cml

Status: Up to date.

See the command line for details.

SiP Layout: Creating die and bump cells

3. Create Die outline.

   a. Select Setup→Application Mode→Symbol Edit.
   b. RMB→Add Component.

      - Set up all the parameters in Options Form.
      - Select Create Component.

   c. Select Component (make sure find filter is set to Comps).
   d. RMB→Add pins (add pin dynamically).

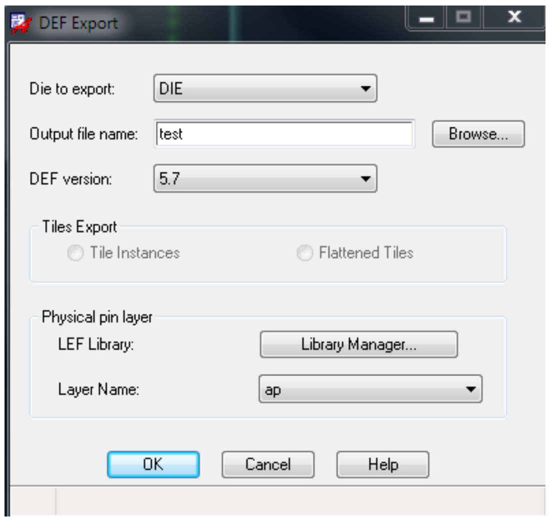
SiP Layout: Exporting initial bump plan to EDIS

4. After initial bump plan, export Die bumps to EDIS. You will use DEF OUT in this flow.

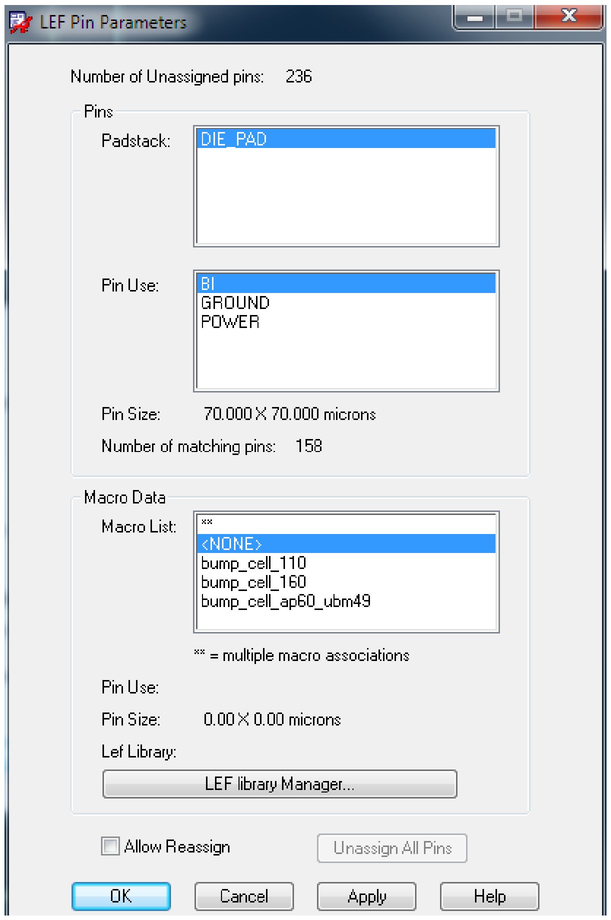
5. Select Export→DEF (Die pins only).

    a. Configure the Physical pin Layer and Layer name for bump cell.
    b. Map the SiP Layout die bump padstack to LEF bump macro.

The following images show a typical setup process:



Die to export: DIE
Output file name: test
DEF version: 5.7
LEF Library:
Layer Name: ap



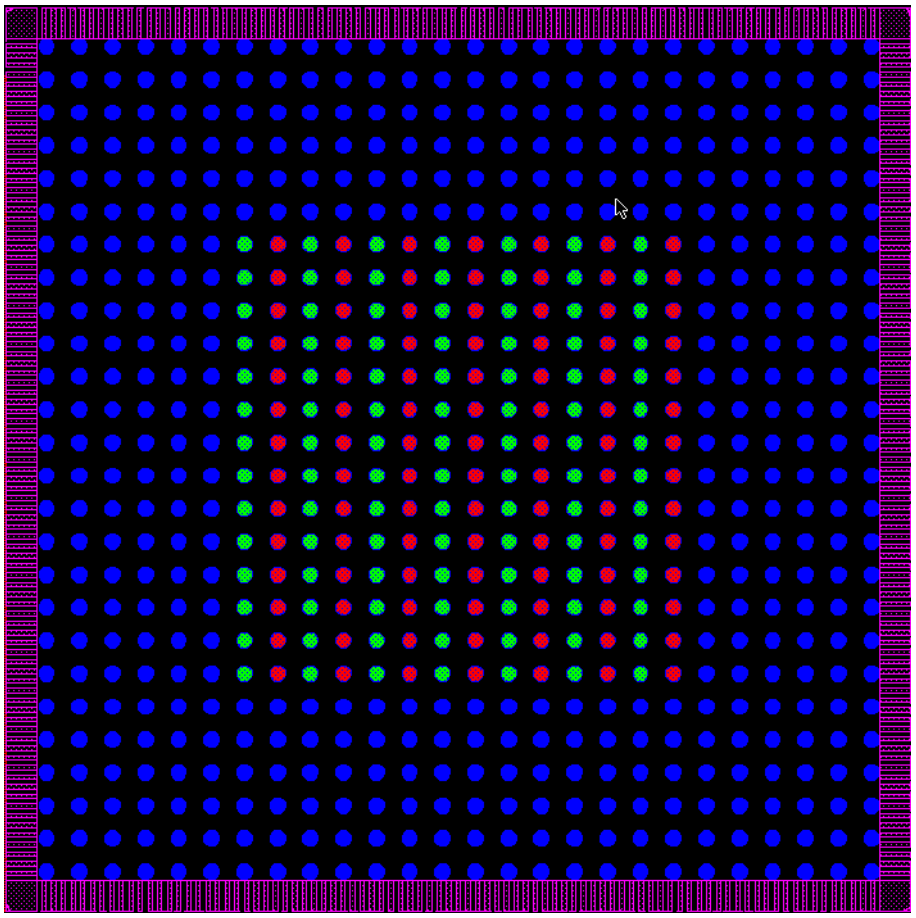
Number of Unassigned pins: 236
Padstack: DIE_PAD
Pin Use: BI
GROUND
POWER
Pin Size: 70.000 X 70.000 microns
Number of matching pins: 158
Macro List: **
<NONE>
bump_cell_110
bump_cell_160
bump_cell_ap60_ubm49
** = multiple macro associations
Pin Use:
Pin Size: 0.00 X 0.00 microns
Lef Library:
Allow Reassign

EDIS: Load initial bump plan

6. Load DEF or areaio file (<design_name>.fp).

   a. To load DEF (via GUI), select File→Load→DEF.

   The following image shows an example of the DEF-based bump array in EDIS:



EDIS: Creating die abstract to exchange with SiP Layout

Once the bumps are in EDIS, it is simple to create a die abstract:

7. Launch EDIS to create die abstract.
8. Open Linux terminal window.
9. cd <your_design_inst_dir>.
10. Encounter and open/restore encounter design <my_design.enc>.
11. To write die abstract from encounter command line:

    a. encounter>write_codesign_abstract <abstract_name>.xda , which is write_codesign_die_abstract my_abstract_name.xda
    b. Use command line “write_codesign_die_abstract –help” to see options.

EDIS⇔SiP: Creating die abstract to exchange with SiP Layout

12. Set db resolution to 3 decimal places, enable Symbol Editor app mode.

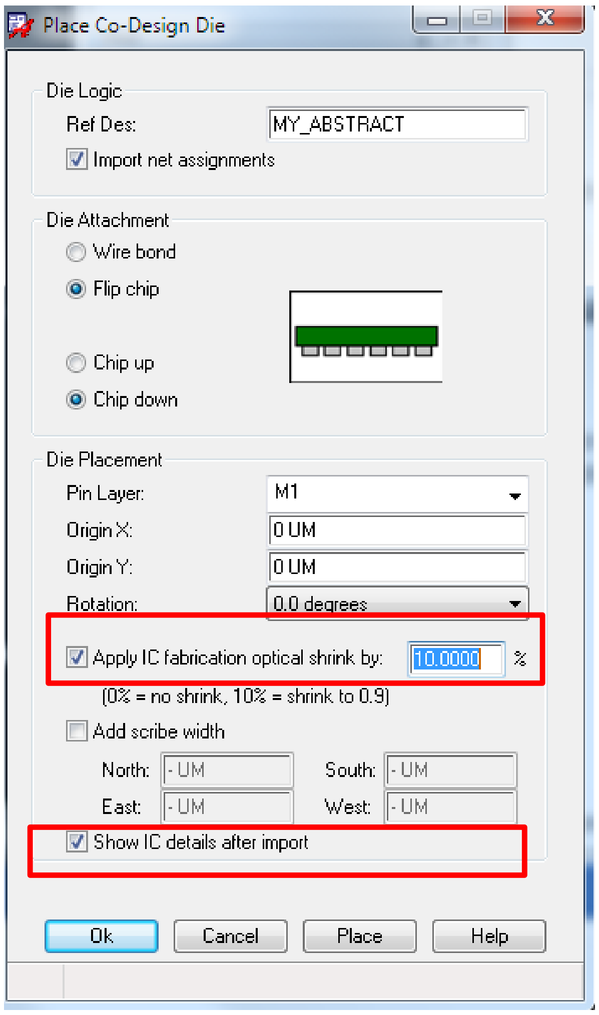
    a. Go to Setup→Design Preferences→Design Parameters Editor→Design Tab.
    b. Set Accuracy to 3.

13. Go to Setup→Application Mode→Symbol Edit.

14. Load Abstract File.

    a. Add→Co-Design Die.
    b. Select <abstract_name>.xda, then click OK.
    c. Set parameters for the Place Co-Design Die Form.
    d. Set Select Show IC details after import and Shrink.

The following image shows a setup form to add Co-design Die:


Place Co-Design Die

Die Logic

Ref Des: MY_ABSTRACT

[✓] Import net assignments

Die Attachment

( ) Wire bond

(●) Flip chip

( ) Chip up

(●) Chip down

Die Placement

Pin Layer: M1

Origin X: 0 UM

Origin Y: 0 UM

Rotation: 0.0 degrees

[✓] Apply IC fabrication optical shrink by: 10.0000 %

(0% = no shrink, 10% = shrink to 0.9)

[ ] Add scribe width

North: - UM South: - UM

East: - UM West: - UM

[✓] Show IC details after import

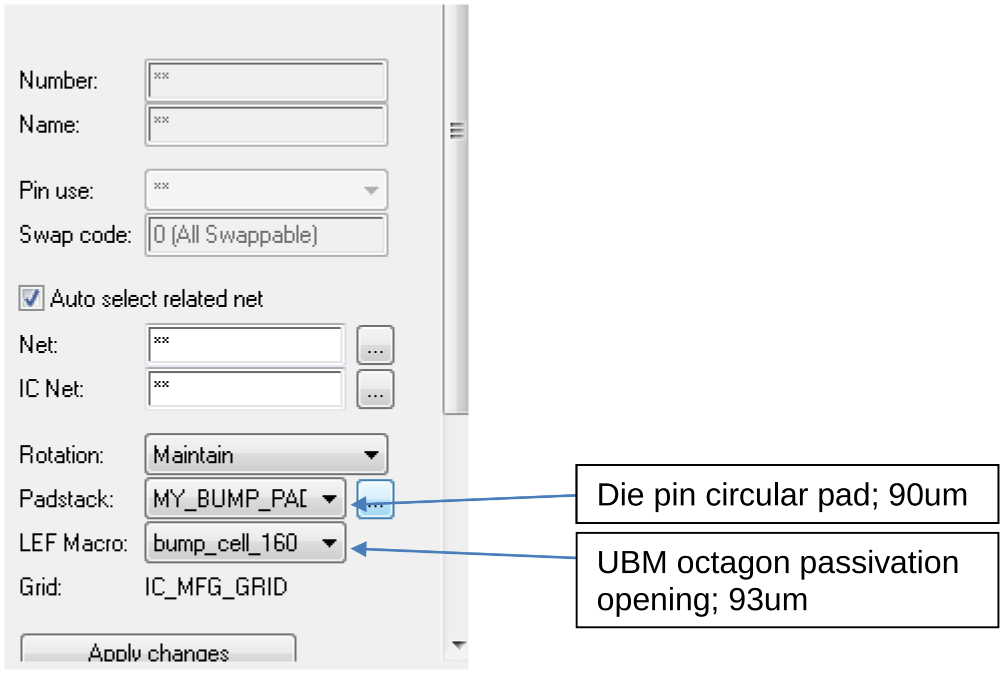
EDIS⇔SiP: Additional die abstract options for UBM and die pin configuration

With abstract, we can intelligently handle both UBM and die pin.

15. UBM (passivation opening) is treated as octagon cell as defined in LEF macro.
16. Die Pins is treated at Circular padstacks as Selected or defined by user, and can be configured taking the following steps:

    a. With Symbol Editor, enable the app mode.
    b. Set Find filter to Pins.
    c. Use LMB and Select Die pins (all or partial).
    d. RMB→Change attributes.

The following image shows an example setup form to add Co-design Die:



Number:	**
Name:	**
Pin use:	**
Swap code:	0 (All Swappable)
☑ Auto select related net	
Net:	**
IC Net:	**
Rotation:	Maintain
Padstack:	MY_BUMP_PAD
LEF Macro:	bump_cell_160
Grid:	IC_MFG_GRID

Die pin circular pad; 90um

UBM octagon passivation opening; 93um

EDIS⇔SiP: Generating SiP Layout data to export to EDIS (as ECO update)

After abstract optimization and other changes, Export the new abstract to EDIS.

17. Go to File→Export→Die Abstract File.
18. Set Abstract writer parameters, and click OK.

EDIS⇔SiP; EDIS importing die abstract (as ECO update)

19. EDIS can import Die Abstract ECOs in two modes:

    a. Mode 1: Graphical mode with abstract design differences checking (you will use this option for the promoted flow):

        I. This mode lets the user to see all the related ECO changes graphically.
        II. EDIS must be running in graphics mode.

    b. Mode 2: Non-graphics mode and bypassing the abstract differences checker:

        I. Abstract ECO will be processed and directly imported into EDIS (unless a checking only is performed).
        II. EDIS will be running in non-graphics mode.

EDIS⇔SiP; EDIS importing die abstract (as ECO update)

For the purposes of this application note, you will only run the die abstract import process using Mode 1, graphical mode with abstract design differences checking.

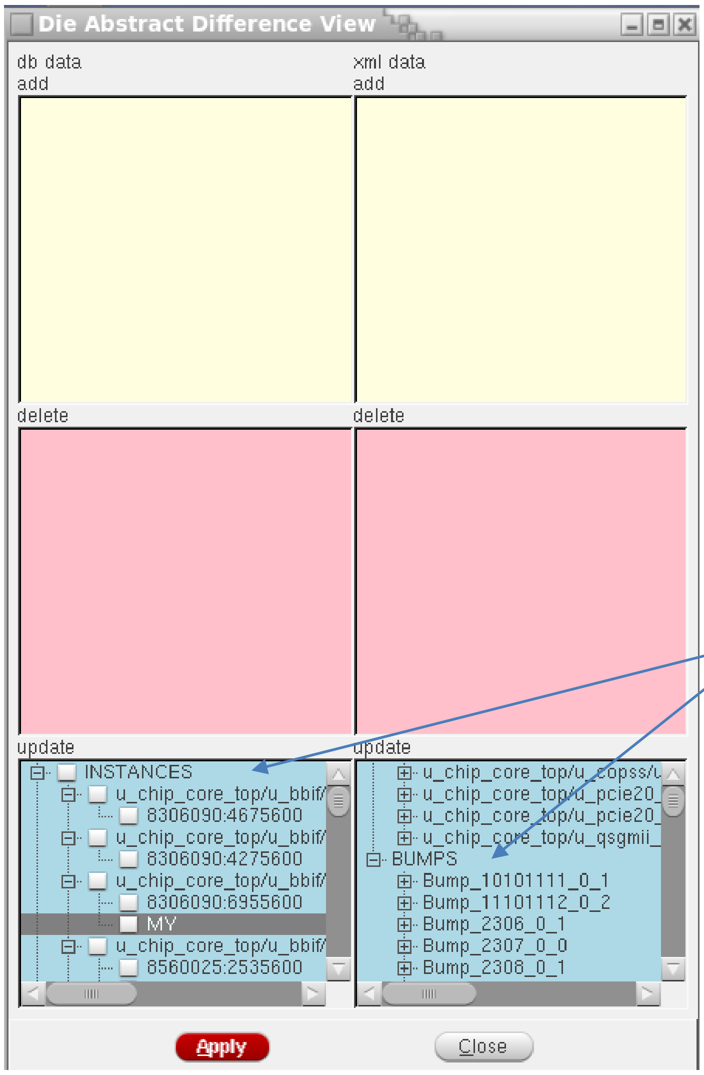
20. To import die abstract using encounter command line:

    a. Launch encounter on <your_design.enc> or restore design if necessary.

    b. Encounter > read_codesign_die_abstract <abstract_name_eco>.xda.

        i. Use read_codesign_die_abstract –help to see options.

The following image shows the abstract difference viewer after EDIS has processed the abstract ECO information coming from the SiP Layout:



Abstract differences shown for the instances and bumps change in the SiP Layout

Summary

This Application Note explains the Pkg and IC design with the concept of early bump planning using the SiP Layout and EDIS environments. Additionally, the die abstract data concept and using DEF as an early bump planning solution is explained as a viable bump plan data exchange mechanism between the SiP Layout and EDIS environment.

Support

Cadence Online Support provides access to support resources, including an extensive knowledge base, access to software updates for Cadence products, and the ability to interact with Cadence Customer Support. Visit *https://support.cadence.com.*

Feedback

Email comments, questions, and suggestions to content_feedback@cadence.com.