A method to plan & generate IO ring based on csv specifications

Manvi Dhawan, Rajeev Singh, Atul Bhargava from STMicroelectronics
Akshita Bansal, Hitesh Marwah, Vishesh Kumar from Cadence

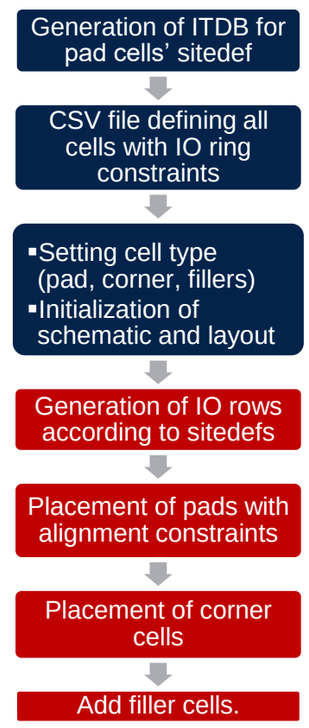
1 Motivation

- For Testchip design IO ring is created for testing IPs and needs to be aligned with the board.
- Specifications for testchip placement are layout driven and available with layout designer.
- Generation of IO ring involves placement of pads, fillers, and corners aligned perfectly.
- Wirebond cells are placed on an outer ring for connection with board.
- To ensure the quality of design it takes many iterations to clean DRC and make the placement even.

2 Flow overview

- Incremental tech file to match placement in rows with cell dimensions.
- Single user input capturing all placement specification.
- Full connectivity established between layout and schematic generated.
- All the spaces filled automatically with filler cells to ensure continuity in placement.
- Customized automated flow over Virtuoso IO planner.

Customized automated flow
Tool features from Virtuoso



3 Incremental technology database

- Incremental database is generated over technology database with additional information about sitedefs for row placement.
- Row height should match the height should match the height of pad cells to be placed in rows.
- To be created for each type of cell placement.

```
scalarSiteDefs(
;(siteDefName   type   width height symInX symInY   symInR0)
;(-----------        ----   ----- ------ ------ ------   -------)
(corner            pad    40.015  40.015  nil    nil    nil)
(it_pad_1            pad    0.1   40.015  nil    nil    nil)
(it_pad_2            pad    0.1   60   nil    nil    nil)
);scalarSiteDefs
);siteDefs
```

Fig1: sample incremental tech file for site definitions

4 Assisted placement setup

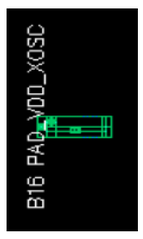
- **Generation of schematic and layout placements.**
  - Layout designer can capture placement specification and cell type in csv format.

#lib_name	cell_name	view_name	relative_spacing	origin	direction	alignment	cellType	comment	Instance_prefix
C28SOI_IO_EXT_CSF_BASIC_EG	LBCORNERCELL_EXT_CSF_FC_LIN	layout	nil	0 0	nil	R0	corner		
C28SOI_IO_EXT_CSF_BASIC_EG	VDD_EXT_CSF_FC_LIN	layout	10nil	up	R270	pad	B16 PAD_VDD_XOSC	PAD_VDD_XOSC	
C28SOI_IO_WBPAD_6U1X2U2X2T8XLB	PAD_WB45SR_6U1X2U2X2T8XLB_V5_10ML	layout	10nil	up	R270	pad			
C28SOI_IO_EXT_CSF_BASIC_EG	GND_EXT_CSF_FC_LIN	layout	150nil	up	R270	pad	B15 PAD_GND_XOSC	PAD_GND_XOSC	
C28SOI_IO_WBPAD_6U1X2U2X2T8XLB	PAD_WB45SR_6U1X2U2X2T8XLB_V5_10ML	layout	10nil	up	R270	pad			
C28SOI_IO_EXT_ALLF_ESDHUB_EG	FILLCELL_ESDHUB_EXT_CSF_FC_LIN	layout	10nil	up	R270	pad			
C28SOI_IO_EXT_CSF_GPIO1V8_FSNFS_LR_0	BDPROGSCRUDQP_EXT_CSF_1V8_FS_FC_LIN	layout	100nil	up	R270	pad	B14 Global_Enable	Global_Enable	
C28SOI_IO_WBPAD_6U1X2U2X2T8XLB	PAD_WB45SR_6U1X2U2X2T8XLB_V5_10ML	layout	10nil	up	R270	pad			

Fig2: Sample input CSV file capturing user specifications

- The cellType for cells are set through skill script for IO planner, based on user specification in csv file.
  - Even if cells are in read only access, cell type is set for placement in current session.
- Additional information for comments and instance name can be captured through CSV.

5 Generation of schematic and layout placement



Names will also capture master cell name for easier debug and analysis in generated Verilog netlist.

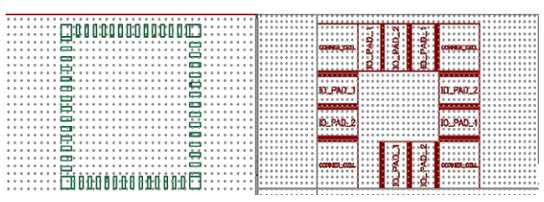
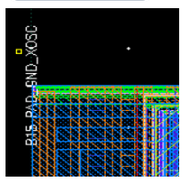


Fig3: Layout and schematic placement

Set cell types → Schematic placement → Layout placement → Correspondence between layout and schematic → IO planner flow

Capture placement information (origin, orientation) and cellType information in csv file.



Additional comments for each instance to be added in canvas for both types of placement.

6 Generation of IO ring

- Automatic IO planner will place pad cells based on their cell types definition as pad cells.
- Cell type is also defined to identify corner cells and filler cells.
- To ensure that order of pad cells is as per user requirement, select the pads and set *alignment* constraint to preserve order.

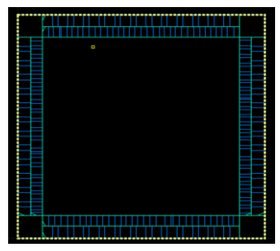
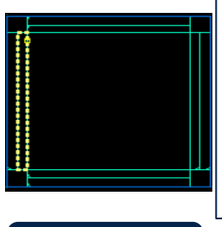


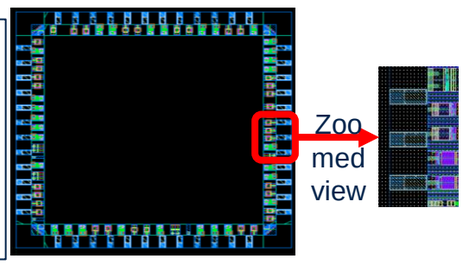
Fig4: IO pads placement on respective sides

7 IO planner flow



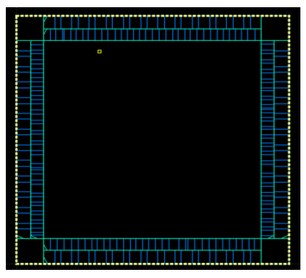
Generation of rows for IO pad placement based on siteDefs defined in ITDB

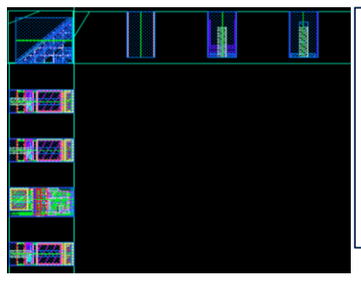
Final placement consists of all the filler cells added to fill gap between pad cells and corner cells



IO Row generation → IO pad placement → Place corner cells → Add fillers

Placement of IO pads in inner row and wirebond cells in outer row according to user specified orientation





Placement of corner cells according to user specified orientation .

8 Summary

- We need to make test chips for all analog IPs.
  - There is **no schematic available.**
  - It takes **3-4 weeks for manually** making the test chip.
  - About **10-15 test chips are delivered** every year.
- The flow proposed provides a semiautomated flow that
  - Saves **40% effort**.
  - **Eliminates manual error prone** placements that are CSV based in the flow.
  - **Quality of the layout improves** as wire bond cells are distributed evenly.
- The flow is **technology independent** and **can be reused** for generation of multiple test chips.