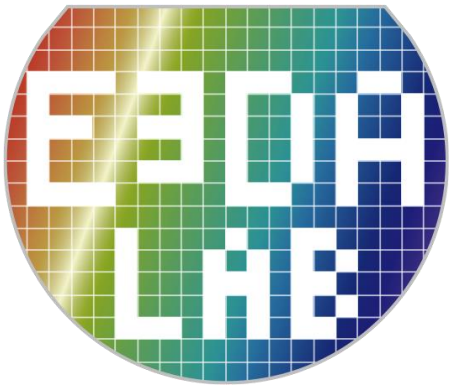

2.5D Holistic Design Flow

Version 1.0

Energy-Efficient Electronics and Design Automation (E3DA) Lab

Department of Computer Science and Computer Engineering (CSCE)

University of Arkansas

Disclaimer of Warranty

The software is provided “AS IS”, without warranty of any kind, express or implied, including but not limited to the warranties of merchantability, fitness for a particular purpose and noninfringement. In no event shall the authors or copyright holders be liable for any claim, damages or other liability, whether in an action of contract, tort or otherwise, arising from, out of or in connection with the software or the use or other dealings in the software.

Table of Contents

Overview

This package contains a shorter version of the design flow presented in the paper [3]. This guide mainly discusses the usage of several scripts used in the flow and how they are integrated in the flow. The scripts used in this package are written using Python version-2.7.5. The scripts depends on following Python packages: numpy, argparse, imp, matplotlib, csv, and sys.

Package Organization

The package contains a design setup for Cadence Innovus v17.17. Within it, there are several subdirectories and files. Table 1 briefly describes the contents of the subdirectories. Though there is a subdirectory named "scripts", the rdl-planner and pin-placement generation scripts are placed in their own subdirectories along with their configuration files. The tcl script "scripts/full_flow_D1.tcl" describes the entire flow for Innovus. If all the required tcl scripts sourced in this flow-script are there, sourcing this scripts after importing the design implements all the design steps of the flow.

Table 1: Package organization

TCPMT-flow-v1	**Contents**
├── **netlist**	Top-level netlist describing the inter-chiplet connections
├── **lib**	Technlogy files and cell libraries
├── **chiplet_macros**	LEF macros of Core-chiplet and Mem-Chiplet
├── **rdl_plan**	RDL planning script and an example RDL plan configuration. All configuration files included.
├── **pin_confs**	Chiplet pin placement generation script and and example pin configuration. All configuration files included.
├── **scripts**	Flow script and external IO routing script
├── **saved_steps_enc**	All the saved steps as encounter database
├── **innovus**	An empty directory to be used as Innovus workspace
├── ***cmsdk_mcu.globals***	Setting file to import the design in Innovus
├── ***cmsdk_mcu.view***	MMMC view definition file, required by the .globals file
└── ***makefile***	Contains the "clean" recipe to clean-up the workspace

Design Flow

To recreate the design found in the "saved_steps_enc/final_D1.enc" execute the steps listed below. There are some expected errors and warnings in some of the following steps. These are listed in a table after each such step. Some messages are truncated for brevity.

1. Start Innovus in the innovus subdirectory. Though the starting directory doesn't matter, this way the workspace remains clean.

---

[3] MD Arafat Kabir, and Yarui Peng, "Holistic Chiplet-Package Co-Optimization for Agile Custom 2.5D Design", *(accepted) IEEE Transactions on Components, Packaging, and Manufacturing Technology*, 2021.

2. Within the Innovus prompt, type "`cd ..`" to come back to the top level directory and follow the steps shows in Fig. 1 to load the design settings using the cmsdk_mcu.globals file.

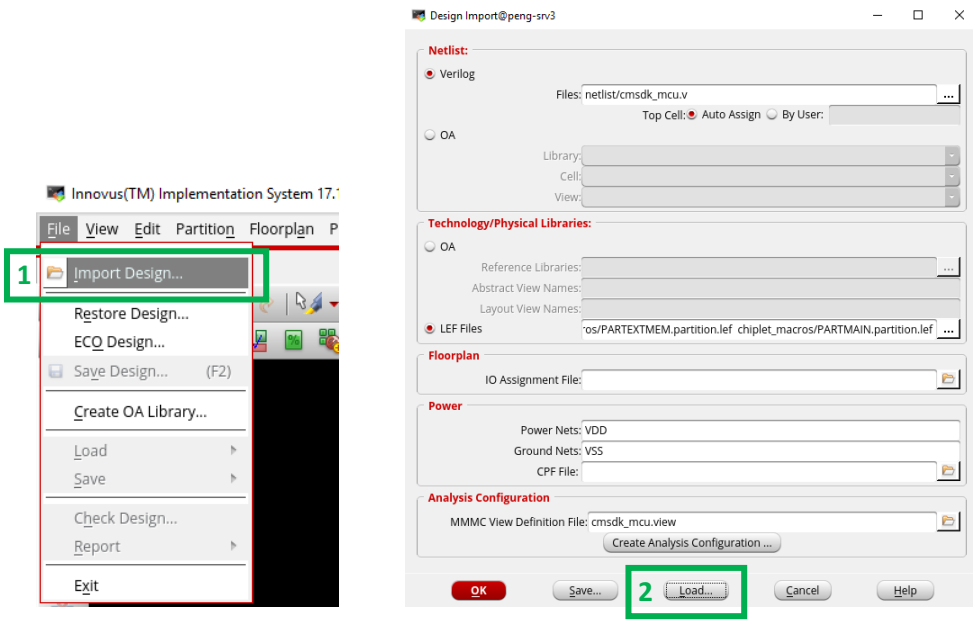

Fig. 1: Loading the design using the cmsdk_mcu.globals file

Error/Warning	Comment
**WARN: (TECHLIB-436): Attribute 'max_fanout' on 'output/inout' pin 'ZN' of cell 'AND2_X4' is not defined in the library.	This warning is coming from the Nangate timing library. There can be several such warning, one for each cell.
**ERROR: (TECHLIB-1346): The attribute 'index_1' defined in group 'ecsm_waveform' on line 393850 is not monotonically increasing for values '0.000041' to '0.000037'. This may lead to undesirable analysis results. The attribute will be ignored.	This error is also coming from the Nangate timing library. There can be a few of this error message.
**WARN: (IMPFP-3961): The techSite 'CoreSite' has no related standard cells in LEF/OA library. Cannot make calculations for this site type unless standard cell models of this type exist in the LEF/OA library.	This warning is due to absence of any standard cell in the top-level design. This warning can be safely ignored.
**WARN: (TCLCMD-1457): Skipped unsupported command: set_units (File netlist/cmsdk_mcu_mod1.sdc, Line 8)	This is coming from the SDC file generated by the synthesis tool (Synopsys DC). This can be safely ignored.
**WARN: (IMPSYC-2): Timing information is not defined for cell PARTMAIN/PARTEXTMEM; Check the timing library (.lib) file and make sure the timing information exists for the cell …	Chiplet blocks are given as black-box LEF macros in this package. This can be safely ignored here.

3. Use the following commands from the “scripts/full_flow_D1.tcl” in Innovus command prompt to setup the package size, place the chiplet macros within the package, and specify the power and ground nets.

```
File: scripts/full_flow_D1.tcl
11 set dsn_num D1
12
13 # Loaded
14 saveDesign saved_steps_enc/loaded_$dsn_num.enc
15
16
17 # Set floorplan size
18 floorPlan -site FreePDK45_38x28_10R_NP_162NW_34O -s 1300 1150 0.0 0.0 0.0 0.0
19 saveDesign saved_steps_enc/floorplan_$dsn_num.enc
20
21
22 # Place chiplet macros
23 placeInstance u_part_extmem  703.19 340.2 R0
24 placeInstance u_part_main    232.94 340.2 R0
25 saveDesign saved_steps_enc/macro_placed_$dsn_num.enc
26
27
28 # Connect Global nets
29 globalNetConnect VDD -type pgpin -pin VDD -inst *
30 globalNetConnect VDD -type tiehi -inst *
31 globalNetConnect VSS -type tielo -inst *
32 globalNetConnect VSS -type pgpin -pin VSS -inst *
33 saveDesign saved_steps_enc/global_net_connected_$dsn_num.enc
```

4. Lines 36-50 of the flow script describes how to use the python scripts written by the author to generate RDL-plan, chiplet pin-placement scripts, and top-level package pin placement scripts. Note that, all the scripts are developed using Python version-2.7.5. So, change the python invocation in your system accordingly.

5. Use the following commands in Innovus command prompt to source the tcl scripts generated by the python scripts to perform package pin placement and inter-chiplet routing. Another script “scripts/ExIO_routing.tcl” is included to automatically route the external package IO pins as manually specified by the author.

```
File: scripts/full_flow_D1.tcl
53 # Place package pins
54 source pin_confs/package_pin_placement.tcl
55 saveDesign saved_steps_enc/top_pins_placed_$dsn_num.enc
56
57
58 # Route RDL wires
59 source rdl_plan/plan_D2/rdl_routing_script_out.tcl
60 saveDesign saved_steps_enc/routed_with_script_$dsn_num.enc
61
62
63 # Route package IO
64 source scripts/ExIO_routing.tcl
65 saveDesign saved_steps_enc/IO_routed_$dsn_num.enc
66
67
68 # Just for record
69 saveDesign saved_steps_enc/final_$dsn_num.enc
```

Error/Warning	Comment
For usability, default behavior of placing the pin at close-by track if location is not on fence and option -side or -edge is not specified is changed. Now by default pin is snapped to closest location on closest edge for the given location. To place the pin inside partition or block boundary add '-side inside' in editPin command. Fixing pin overlap …	This message appears after executing the command at line 54. This message is safe to ignore.

6. To clean-up the workspace, use the "clean" recipe in the makefile as shown below. This will remove all the files generated by the scripts written by the author.

```
$ make clean
```

Note: All these steps are already saved in the subdirectory "saved_steps_enc" in the package. To load a saved design, follow the steps 1-2 and then source any saved encounter design from the "saved_steps_enc" subdirectory. For example, to load the design saved at line 60 use the following command,

```
innovus 2> source saved_steps_enc/routed_with_script_D1.enc
```

Scripts Usage

This section briefly describes the usage of the scripts written by the author and the configuration files used with them.

gen_package_pins.py

This script is located at "pin_confs/gen_package_pins.py." This is the simplest script without any options and configuration file. This script generates a script for Innovus, which is use for package

pin placement. The variables like die_width, die_height, etc. at the top of the script are set up based on the design included in the package. To use the script, just invoke it using Python-2.7 interpreter in your shell,

```
$ python gen_package_pins.py
```

gen_rdl_plan.py

This script is located at "rdl_plan/plan_D2/gen_rdl_plan.py." This script implements the RDL-planning algorithm described in the paper [1]. This script takes a single configuration file as the command line argument. This configuration file specifies other files to read design related information like timing slacks and pre-placed pins. An example configuration file is located at "rdl_plan/plan_D2/rdl_plan_settings.conf." To use the script, invoke it using Python-2.7 interpreter in your shell specifying the configuration file as an argument,

```
$ python gen_rdl_plan.py <configuration.conf>
```

The included configuration file provides the RDL-planning script with the design information necessary to finish the planning as described in the paper [1]. Following illustration uses comments to explain each option used in the provided configuration file.

File: rdl_plan/plan_D2/rdl_plan_settings.conf

```
 1 # Layout configuration
 2 pitch_unit = 20                   # package routing track pitch in micron
 3 chiplet_origin = (663.065, 355.2) # (x,y) of the bottom mid-point between the chiplets
 4 chiplet_distance = 110.25         # distance between the chiplets (center of pins)
 5 layer_names = ['rdl1','rdl2','rdl3']    # package layer names
 6 via_names = ['R2_R1_via', 'R3_R2_via']  # via names between RDLs (not used)
 7 pin_size = (10.0, 10.0)                 # (width, height)
 8 net_slack_csv = "pkg_wire_slacks.csv"   # a CSV containing following fields,
 9                                         #    (net-name, ch1 pin, ch2 pin, slack)
10
11 # Left chiplet configurations
12 Lchip_name = "PARTMAIN"  # name of the chiplet module
13 Lchip_row_cnt = 15       # number of rows of the pin grid
14 Lchip_col_cnt = 10       # number of columns of the pin grid
15 Lchip_pin_row_pitch = 2  # number of tracks
16 Lchip_pin_col_pitch = 2  # number of tracks
17 Lchip_layer_cnt = 3      # number of RDL layers to use
18 #Lchip_name_csv = 'pin_conf_PARTMAIN_D4.csv'
19 #Lchip_pin_to_net_csv = 'PARTMAIN_pin_to_net_map.csv'
20
21
22 # Right chiplet configurations
23 Rchip_name = "PARTEXTMEM"
24 Rchip_row_cnt = 12
25 Rchip_col_cnt = 9
26 Rchip_pin_row_pitch = 2  # number of tracks
27 Rchip_pin_col_pitch = 2  # number of tracks
28 Rchip_layer_cnt = 3
29 Rchip_pin_name_csv = 'pre_placed_pins_PARTEXTMEM.csv'   # for pre-placed pins
30 #Rchip_pin_to_net_csv = 'PARTEXTMEM_pin_to_net_map.csv'
```

gen_partition_pins.py

This script is located at "pin_confs/gen_partition_pins.py." This script takes two positional arguments and one optional argument. The first positional argument needs to be a configuration file describing the required pin-arrangement for the chiplet. The second positional argument needs to be a CSV file containing the pin names of the chiplet, arranged in the specified pin-grid. The optional argument is specified using the "-p" option, which is used as the prefix for the names of the generated files. The script generates two files, a tcl script to be used by Innovus to place the chiplet partition pins and a DIE.txt file (not necessary for this flow). To use the script, invoke it using Python-2.7 interpreter in your shell using the following syntax,

```
$ python gen_partition_pins.py <configuration.conf> <pin-names.csv> [-p name_prefix]
```

Following illustration describes the configuration file used for generating the pin-placement script for the core-chiplet. Each option is described in the inline comments.

File: pin_confs/pin_conf_D2/pin_settings_PARTMAIN.conf

```
grid_width = 370          # Pin-grid width in micron
grid_height = 570         # Pin-grid height in micron
offset_x = 10             # x-axis offset from the lower-left corner of the chiplet
offset_y = 10             # y-axis offset from the lower-left corner of the chiplet
row_num = 15              # number of rows in the pin-grid
col_num = 10              # number of columns in the pin-grid
pin_width = 10.0          # pin-width in micron
pin_height = 10.0         # pin-height in micron

power_name = 'VDD'
ground_name = 'VSS'
cell_name = 'PARTMAIN'
layer = 'rdl1'                  # layer to place the pins on
cell_origin = (232.94, 340.2)   # (x,y) of lower-left corner of the chiplet on package
```

Note that, the pin-placement scripts generated by this script is not directly used in this shorter version of the flow. However, the chiplet-macros are created by the author using the tcl scripts generated by this python script.

Related Publications

This section contains a list of all the current publications related to the 2.5D holistic design flow as of the date of this document.

[1] MD Arafat Kabir, and Yarui Peng, "Chiplet-Package Co-Design For 2.5D Systems Using Standard ASIC CAD Tools", in *Proc. Asia and South Pacific Design Automation Conference*, pp. 351–356, Jan 2020.

[2] MD Arafat Kabir, and Yarui Peng, "Holistic 2.5D Chiplet Design Flow: A 65nm Shared-Block Microcontroller Case Study", in *Proc. IEEE International System-on-Chip Conference*, Oct 2020.

[3] MD Arafat Kabir, and Yarui Peng, "Holistic Chiplet-Package Co-Optimization for Agile Custom 2.5D Design", *(accepted) IEEE Transactions on Components, Packaging, and Manufacturing Technology*, 2021.

Userful Links

- **Release website:** https://e3da.csce.uark.edu/release/2.5D_Design_Flow/
- **Publication website:** https://e3da.csce.uark.edu/pub/

Authors

This flow is still under development. A brief introduction to the authors are as follows.

- **MD Arafat Kabir**
  Ph. D. Student
  Computer Science and Computer Engineering Department
  University of Arkansas, Fayetteville, AR, USA
  Email: makabir@uark.edu

- **Dr. Yarui Peng**
  Assistant Professor
  Computer Science and Computer Engineering Department
  University of Arkansas, Fayetteville, AR, USA
  Phone: 479-575-6043
  Email: yrpeng@uark.edu