University of Texas at Dallas
Department of Electrical Engineering
EEDG 6306 - Application Specific Integrated Circuit Design
Synopsys Tools Tutorial

By
Zhaori Bi
Minghua Li

Fall 2014
# Table of Contents

Chapter 2 IC Compiler ........................................................................................................3

Introduction..........................................................................................................................3
  Floorplanning.....................................................................................................................3
  Placement............................................................................................................................7
  Routing...............................................................................................................................7

Tutorial Example ..................................................................................................................9

Setup ....................................................................................................................................9
  1. Log into engnx server and source Synopsys. ...............................................................9
  2. Open ICC and its GUI. ..................................................................................................9
  3. Create design library. .................................................................................................9
  4. Create pads for MSDAP I/O pins ...............................................................................9
  5. Read in gate netlist and sdc file ................................................................................10
  6. Create pads for VDD VSS Corner cells .....................................................................11
  7. Specify Pad Cell Locations .......................................................................................11
  8. Initialize the floorplan ...............................................................................................12
  9. Insert Pad filler ..........................................................................................................13
  10. Connect Power and Ground pins ..........................................................................13
  11. Create Pad Rings ......................................................................................................13
  12. Create VDD/VSS Rings ............................................................................................14
  13. Create Power Strap (VDD,VSS) ..............................................................................15
  14. Placement ..................................................................................................................16
  15. Clock Tree Synthesis ...............................................................................................17
  16. Routing .....................................................................................................................19
  17. Post Routing optimization .......................................................................................19
  18. RC Extraction ..........................................................................................................21
  19. Generate output files for Post-Layout Timing Verification .......................................21
  20. Save your design .....................................................................................................21

Type in: ..............................................................................................................................21
Chapter 2 IC Compiler

Introduction
IC Compiler is the physical implementation tool from Synopsys. The physical implementation step in the ASIC flow consists of:

1. Floorplanning
2. Placement
3. Routing

This tutorial will cover each of them in following sections.

Floorplanning
The main objectives of floorplanning are to minimize:

1. Area
2. Timing (delay)

During floorplanning, the following are done:

1. The size of the chip is estimated.
2. The various blocks in the design, are arranged on the chip.
3. Pin assignment is done.
4. The I/O and Power planning are done.
5. The type of clock distribution is decided.
Floorplanning is a major step in the physical implementation process. The final timing, quality of the chip depends on the floorplan design. The three basic elements of chip are:

1. **Standard Cells**: The design is made up of standard cells.

2. **I/O cells**: These cells are used to help signals interact to and from the chip.

3. **Macros (Memories)**: To store information using sequential elements takes up lot of area. A single flip flop could take up 15 to 20 transistors to store one bit. Therefore special memory elements are used which store the data efficiently and also do not occupy much space on the chip comparatively. These memory cells are called macros. Examples of memory cells include 6T SRAM (Static Dynamic Access Memory), DRAM (Dynamic Random Access Memory) etc.

The above figure shows a basic floorplan. The following is the basic floorplanning steps (and terminology):

1. **Aspect ratio (AR)**: It is defines as the ratio of the width and length of the chip. From the figure, we can say that aspect ratio is x/y. In essence, it is the shape of the rectangle used for placement of the cells, blocks. The aspect ratio should take into account the number of routing resources available. If there are more horizontal layers, then the rectangle should be long and width should be small and vice versa if there are more vertical layers in the design. Normally, METAL1 is used up by the standard cells. Usually, odd numbered layers are horizontal layers and even numbered layers are vertical. So:
   a. For a 5 layer design, AR = 2/2 = 1.
   b. For a 6 layer design, AR = 2/3 = 0.66

2. **Concept of Rows**: The standard cells in the design are placed in rows. All the rows have equal height and spacing between them. The width of the rows can vary. The standard cells in the rows get the power and ground connection from VDD and VSS rails which are placed on either side of the cell rows. Sometimes, the technology allows the rows to be flipped or abutted, so that they can share the power and ground rails.

3. **Core**: Core is defined as the inner block, which contains the standard cells and macros. There is another outer block which covers the inner block. The I/O pins are placed on the outer block.

4. **Power Planning**: Signals flow into and out of the chip, and for the chip to work, we need to supply power. A power ring is designed around the core. The power ring contains both the VDD and VSS rings. Once the ring is placed, a power mesh is designed such that the power reaches all the cells easily. The power mesh is nothing but horizontal and vertical lines on the chip. One needs to assign the metal layers through which you want the power to be routed. During power planning, the VDD and VSS rails also have to be defined.

5. **I/O Placement**: There are two types of I/Os.
   a. Chip I/O: The chip contains I/O pins. The chip consists of the core, which contains all the standard cells, blocks. The chip I/O placement consists of the placement of I/O pins and also the I/O pads. The placement of these I/O pads depends on the type of packaging also. (Refer document: Packaging)
b. Block I/O: The core contains several blocks. Each block contains the Block I/O pins which communicate with other blocks, cells in the chip. This placement of pins can be optimized.

6. Pin Placement: Pin Placement is an important step in floorplanning. You may not know where to place the pins initially, but later on when you get a better idea, the pin placement can be done based on timing, congestion and utilization of the chip.

Pin Placement in Macros: it uses up m3 layers most of the time, so the macro needs to be placed logically. The logical way is to put the macros near the boundary. If there is no connectivity between the macro pins and the boundary, then move it to another location.

7. Concept of Utilization: Utilization is defined as the percentage of the area that has been utilized in the chip. In the initial stages of the floorplan design, if the size of the chip is unknown, then the starting point of the floorplan design is utilization. There are three different kinds of utilizations.

a. Chip Level utilization: it is the ratio of the area of standard cells, macros and the pad cells with respect to area of chip.

$$\frac{\text{Area (Standard Cells)} + \text{Area (Macros)} + \text{Area (Pad Cells)}}{\text{Area (Chip)} - \text{Area (sub floorplan)}}$$

b. Floorplan Utilization: it is defined as the ratio of the area of standard cells, macros, and the pad cells to the area of the chip minus the area of the sub floorplan.

$$\frac{\text{Area (Standard Cells)} + \text{Area (Macros)} + \text{Area (Pad Cells)}}{\text{Area (Chip)} - \text{Area (sub floorplan)}}$$

c. Cell Row Utilization: It is defined as the ratio of the area of the standard cells to the area of the chip minus the area of the macros and area of blockages.

$$\frac{\text{Area (Standard Cells)}}{\text{Area (Chip)} - \text{Area (Macro)} - \text{Area (Region Blockages)}}$$

8. Macro Placement: As a part of floorplanning, initial placement of the macros in the core is performed. Depending on how the macros are placed, the tool places the standard cells in the core. If two macros are close together, it is advisable to put placement blockages in that area. This is done to prevent the tool from putting the standard cells in the small spaces between the macros, to avoid congestion. Few of the different kinds of placement blockages are:

a. Standard Cell Blockage: The tool does not put any standard cells in the area specified by the standard cell blockage.

b. Non Buffer Blockage: The tool can place only buffers in the area specified by the Non Buffer Blockage.

c. Blockages below power lines: It is advisable to create blockages under power lines, so that they do not cause congestion problems later. After routing, if you see an area in the design with a lot of DRC violations, place small chunks of placement blockages to ease congestion.
After Floorplanning is complete, check for DRC (Design Rule check) violations. Most of the pre-route violations are not removed by the tool. They have to be fixed manually. 

I/O Cells in the Floorplan: The I/O cells are nothing but the cells which interact in between the blocks outside of the chip and to the internal blocks of the chip. In a floorplan these I/O cells are placed in between the inner ring (core) and the outer ring (chip boundary). These I/O cells are responsible for providing voltage to the cells in the core. For example: the voltage inside the chip for 90nm technology is about 1.2 Volts. The regulator supplies the voltage to the chip (Normally around 5.5V, 3.3V etc). 

The next question which comes to mind is that why is the voltage higher than the voltage inside the chip? The regulator is basically placed on the board. It supplies voltage to different other chips on board. There is lot of resistances and capacitances present on the board. Due to this, the voltage needs to be higher. If the voltage outside is what actually the chip need inside, then the standard cells inside of the chip get less voltage than they actually need and the chip may not run at all. So now the next question is how the chips can communicate between different voltages? The answer lies in the I/O cells. These I/O cells are nothing but Level Shifters. Level Shifters are nothing but which convert the voltage from one level to another. 

The Input I/O cells reduce the voltage coming from the outside to that of the voltage needed inside the chip and output I/O cells increase the voltage which is needed outside of the chip. The I/O cells acts like a buffer as well as a level shifter.
Placement
Placement is a step in the Physical Implementation process where the standard cells location is defined to a particular position in a row. Space is set aside for interconnect to each logic/standard cell. After placement, we can see the accurate estimates of the capacitive loads of each standard cell must drive. The tool places these cells based on the algorithms which it uses internally. It is a process of placing the design cells in the floorplan in the most optimal way.

What does the Placement Algorithm want to optimize?

The main of the placement algorithm is

a. Making the chip as dense as possible (Area Constraint)
b. Minimize the total wire length (reduce the length for critical nets)
c. The number of horizontal/vertical wire segments crossing a line.

Constraints for doing the above are:

a. The placement should be routable (no cell overlaps; no density overflow).
b. Timing constraints are met

There are different algorithms to do placement. The most popular ones are as follows:

1. Constructive algorithms: This type of algorithm uses a set of rules to arrive at the optimized placement. Example: Cluster growth, min cut, etc.

2. Iterative algorithms: Intermediate placements are modified in an attempt to improve the cost function. It uses an already constructed placement initially and iterates on that to get a better placement. Example: Force-directed method, etc

3. Nondeterministic approaches: simulated annealing, genetic algorithm, etc.

Routing
After the floorplanning and placement steps in the design, routing needs to be done. Routing is nothing but connecting the various blocks in the chip with one another. Until now, the blocks were only just placed on the chip. Routing also is split into two steps

1. Global routing: It basically plans the overall connections between all the blocks and the nets. Its main aim is to minimize the total interconnect length, minimize the critical path delay. It determines the track assignments for each interconnect.

a. The chip is divided into small blocks. These small blocks are called routing bins. The size of the routing bin depends on the algorithm the tool uses. Each routing bin is also called a gcell. The size of this gcell dependson the tool. Each gcell has a finite number of horizontal and vertical tracks. Global routing assigns nets to specific gcells but it does not define the specific tracks for each of them. The global router connects two different gcels from the centre point of each gcell.
b. Track Assignment: The Global router keeps track of how many interconnections are going in each of direction. This is nothing but the routing demand. The number of routing layers that are available depend on the design and also, if the die size is more, the greater the routing tracks. Each routing layer has a minimum width spacing rule, and its own routing capacity. For Example: For a 5 metal layer design, if Metal 1, 4, 5 are partially up for inter-cell connections, pin, VDD, VSS connections, the only layers which are routable 100% are Metal2 and Metal3. So if the routing demand goes over the routing supply, it causes Congestion. Congestion leads to DRC errors and slow runtime.

2. Detailed Routing: In this step, the actual connection between all the nets takes place. It creates the actual via and metal connections. The main objective of detailed routing is to minimize the total area, wire length, delay in the critical paths. It specifies the specific tracks for the interconnection; each layer has its own routing grid, rules. During the final routing, the width, layer, and exact location of the interconnection are decided. After detailed routing is complete, the exact length and the position of each interconnect for every net in the design is known. The parasitic capacitance, resistance can now is extracted to determine the actual delays in the design. The parasitic extraction is done by extraction tools. This information is back annotated and the timing of the design is now calculated using the actual delays by the Static Timing Analysis Tool. After timing is met and all other verification is performed such as LVS, etc, the design is sent to the foundry to manufacture the chip.
Tutorial Example

Setup
1. Gate-level netlist, which you can find it in your dc_out folder named as MSDAP_NETLIST.v

2. Standard cell library. We will provide you all necessary path in script files which will be covered in following sections.

3. Design constrains, which you can find it in your dc_out folder named as MSDAP.sdc

1. Log into engnx server and source Synopsys.

When the command prompt, type in:

{engnx03:~} . /proj/cad/startup/profile.synopsys

*Note that there is a space between ‘.’ and ‘/’

This command enable you open synopsis tools. You should make sure this step is performed before opening any synopsis tools.

2. Open ICC and its GUI.

Type in:

{engnx03:~} icc_shell

icc_shell> gui_start

3. Create design library.

Copy icc_lib.tcl to your ICC working directory and source it on icc_gui:

4. Create pads for MSDAP I/O pins.

Copy the contents of Chip.v file to the last line of your netlist file. This fragment of Verilog code put your I/O pins to the corresponding I/O pads.
5. Read in gate netlist and sdc file

Copy `icc_read.tcl` to your ICC working directory and source it on icc_gui:

```
source icc_read.tcl
```

Now you will see a layout window, which contains the layout information as shown below. You can see all the cells in the design at the bottom, since we have not initialized the floorplan yet or done any placement.
6. Create pads for VDD VSS Corner cells

Copy `create_cell.tcl` to your ICC working directory and source it. This script creates the physical-only pad cells (VDD/VSS, Corner cells) that are not part of the synthesized netlist.

```
log history

icc_shell> source create_cell.tcl

Ready
```

7. Specify Pad Cell Locations

Copy `io.tdf` file to your ICC working directory and type in:

```
read_pin_pad_physical_constraints io.tdf
```

```
log history

icc_shell> read_pin_pad_physical_constraints io.tdf

Executing command 'source icc_read.tcl' ...
```
This script file specifies the location of each port as your specification. Now you are ready to move to floorplan stage.

8. Initialize the floorplan

Copy floorplan.tcl to your ICC working directory and source it:

```
Log History
Icc_shell> source floorplan.tcl
Ready
```

Modify the utilization, the margin between I/O and core to fit your design.

```
initialize_floorplan -core_utilization 0.6 -start_first_row -left_io2core 20.0 -bottom_io2core 20.0 -right_io2core 20.0 -top_io2core 20.0 -pin_snap
```

Now you have the layout similar to the following picture.
9. Insert Pad filler

Type in:

```
insert_pad_filler -cell "PFILL50 PFILL5 PFILL20 PFILL2 PFILL10 PFILL1 PFILL01 PFILL001" \
   -overlap_cell "PFILL001"
```

10. Connect Power and Ground pins

Type in:

```
derive_pg_connection -power_net VDD -ground_net VSS
```

```
derive_pg_connection -power_net VDD -ground_net VSS -tie
```

11. Create Pad Rings

Type in:

```
create_pad_rings -create_pg -route_pins_on_layer METAL6
```
12. Create VDD/VSS Rings

Type in:

```bash
##Create VSS ring
create_rectangular_rings -nets {VSS}\
-left_offset 5 -left_segment_layer METAL5 -left_segment_width 2.0 -extend_ll -extend_lh\n-right_offset 5 -right_segment_layer METAL5 -right_segment_width 2.0 -extend_rl -extend_rh\n-bottom_offset 5 -bottom_segment_layer METAL6 -bottom_segment_width 2.0 -extend_bl -extend_bh\n-top_offset 5 -top_segment_layer METAL6 -top_segment_width 2.0 -extend_tl -extend_th
##Create VDD ring
create_rectangular_rings -nets {VDD}\
-left_offset 12 -left_segment_layer METAL5 -left_segment_width 2.0 -extend_ll -extend_lh\n-right_offset 12 -right_segment_layer METAL5 -right_segment_width 2.0 -extend_rl -extend_rh\n-bottom_offset 12 -bottom_segment_layer METAL6 -bottom_segment_width 2.0 -extend_bl -extend_bh\n-top_offset 12 -top_segment_layer METAL6 -top_segment_width 2.0 -extend_tl -extend_th
```

Modify the space between the core to rings to fit your design. The design parameter is shown as following picture:
13. Create Power Strap (VDD, VSS)

Modify the number of straps and position to fit your design:

```plaintext
# Below commands connect power strap for VDD
create_power_straps -direction vertical -start_at 271.25 -num_placement_strap 51 -increment_x_or_y 30 -nets {VDD} -layer METAL5 -width 1
# Below commands connect power strap for VSS
create_power_straps -direction vertical -start_at 286.25 -num_placement_strap 50 -increment_x_or_y 30 -nets {VSS} -layer METAL5 -width 1
```
14. Placement

Click Placement->Core Placement and Optimization …

Check the optimization options you interest to perform and click ok.

Generate utilization report by typing in:

`report_placement_utilization > icc_rpt/util.rpt`

Make sure there is no congestion issue on your design. If you got the error report try to fix it by following ways:

a. Try high effort optimization.

b. Adjust your floor plan utilization to a small value.

After placement is done your layout will similar with the following picture
15. Clock Tree Synthesis

Click Clock->Core CTS and Optimization

Check the optimization options you interest to perform and click ok.
Generate timing report by typing in:

\[
\text{set_operating_conditions} \text{-min_library ff}_\text{1v98}_0c \text{-min ff}_\text{1v98}_0c \text{-max_library ss}_\text{1v62}_125c \text{-max ss}_\text{1v62}_125c
\]

\[
\text{report_timing} \text{-max_paths 20 -delay max} > \text{icc_rpt/setup.rpt}
\]

\[
\text{set_operating_conditions} \text{-min_library ff}_\text{1v98}_0c \text{-min ff}_\text{1v98}_0c \text{-max_library ss}_\text{1v62}_125c \text{-max ss}_\text{1v62}_125c
\]

\[
\text{report_timing} \text{-max_paths 20 -delay min} > \text{icc_rpt/hold.rpt}
\]

Make sure there is no slack (VIOLATED) in your report. Now you are ready to routing.
16. Routing

Click Route -> Core Routing and Optimization

You can select various options, if you want all the routing steps in one go, or do global routing first, and then detail, and then optimization steps. It is up to you.

After the routing is complete check the shell window report. Make sure there is no DRC violating.

17. Post Routing optimization

Click Route -> Verify Route

Click ok
Check the result on shell window makes sure the results are clean. If it is not you may perform incremental route.
18. RC Extraction

Click Route -> Extract RC

![Extract RC dialog box]

Click ok.

Generate power report

`report_power > icc_rpt/power.rpt`

19. Generate output files for Post-Layout Timing Verification

a. Generate parasitic file

`write_parasitics -output ./icc_out/MSDAP.spef -format SPEF`

b. Generate Standard Delay Format (SDF) back-annotation file

`write_sdf ./icc_out/MSDAP.sdf`

c. Generate Synopsys Design Constraints (SDC) file

`write_sdc ./icc_out/MSDAP.sdc`

d. Generate Verilog file for current design (layout)

`write_verilog ./icc_out/MSDAP.v`

20. Save your design

Type in:

`save_mw_cel --as MSDAP`