

Sharif University of Technology Department of Computer Engineering

Advanced VLSI Design Course Project Physical Design and Layout Flow Using Cadence for Synthesized Digital Circuits

Mahbod Afarin December 2015

1- Introduction and Problem Definition

In project 2, we used the Design Compiler (DC) tool to convert high-level code into a **netlist**, ultimately producing a **gate-level circuit**. Essentially, we created a mapping between the **technology cells** and the **circuit we had**. In this project, we want to provide our circuit to the **SOC Encounter** software so that it performs **Placement & Routing (P&R)**. The output can then be sent to a **fabrication facility**, which can manufacture the IC for us. In this task, we will **not optimize the circuit using Desing Compiler**, because optimization may lead to some gates being removed, and as a result, the final circuit might contain wires that are **not connected anywhere**.

2- Project Steps

It is important to always open **SOC Encounter** from the **directory of your design**.

We navigate to the appropriate path and provide the **netlist**, which was generated as an output from DC. Then, we enter the **top module name**, and we also add the **.sdk file**, which defines the timing constraints.

The next file is the **.lef file**, which contains the **physical technology** information. This tells the software things like how many layers the standard cells have, the pitch, and so on.

For the LEF file, we add two files:

- One is the **standard cell LEF file**
- The other is the **antenna effect file**

The **antenna effect** refers to the issue that arises when routing becomes dense: each wire behaves like an antenna, and its capacitance can introduce **noise** into the circuit. To manage this, we use the **antenna library**, which contains antenna-related specs for each cell.

Next, under the **Advanced** tab and the **Power** section, we define the **global VDD and GND** for the circuit. All standard cells must have their VDD and VSS specified. Since VDD and GND come from outside the chip, we must explicitly define the global power and ground for our design.

Then, as shown in the figure, we go to **Specify Floorplan**. We activate the option "**Die size by**" and assign an area larger than the default. The default area is minimal, so we increase it to ensure sufficient space.

Specify Eleorplan	
Basic Advanced	
Design Dimensions	
Specify By: 🔶 Size 🗸 Die/IO/Core Coordinates	
✓ Core Size by: ◆ Aspect Ratio: Ratio (H/V)	V): 0.99983772
Core Utilization	0.548141
🕹 Cell Utilization	0.545325
🔷 Dimension: Wid	th: 369.74
Heig	ht: 369.68
 Die Size by: Widt 	th: 405.0
Heig	ht: 405.0
Core Margins by: 🗸 Core to IO Boundary	
Core to Die Boundary	
Core to Left: 30 Core to To	ip: 30
Core to Right: 30 Core to Botto	m: 30
IO Box Calculation Use: 💊 Max IO Height 🔶	Min IO Height
🔰 🖌 Floorplan Origin at: 🔷 Lower Left Corner	Center
	Unit: Micron
<u>QK</u> <u>Apply</u> <u>Cancel</u>	Help

Next, to connect the core to **VSS** and **VDD**, two complete **power rings** (one for VSS and one for VDD) are drawn around the core. The existing VDD and VSS pins in the core are then connected to these rings. This approach helps to **reduce power consumption**, **increase noise immunity**, and **optimize routing**. Therefore, in the **"core to left"** and **"core to right"** sections, we specify a spacing value to provide room for the power rings. The result is as shown in the figure below.



As shown in the figure above, a **spacing** has been created between the **core** and the **die**. Then, as shown in the next figure, we go to the **"Connect Global Net"** section and connect the **power** and **ground** cells to the **global VDD** and **GND**.

The **tie-high** concept means that in digital circuits, when we want to assign a logic high ('1') to a node, we **do not connect it directly to VDD**. This is because if **noise** appears on VDD, it can affect the connected node as well. To prevent this, two special cells called **tie-high** and **tie-low** are used:

- tie-high generates a logic high ('1') that is resistant to noise
- tie-low generates a logic low ('0') that is also noise-resistant

Y	Global Net Connections	= = ×
Connection List Vdd:PIN:".vdd:All vdd:TIEHI:".vdd:All vss:PIN.".vss:All vss:TIELO:".vss:All vss:NET:vss:All	Global Net Connection Connect ✓ ✓ Pin ✓ Tie High ✓ Tie Low Instance Basename: * Pin Name(s): V\$S ◆ Net Basename: V\$S ◆ Scope ✓ Single Instance: ✓ Under Module: ✓ Under Power Domain: ✓ Under Region: Ibx: 0.0 Ily: 0.0 urx: 0.0 ♦ Apply All	
	✓ Under Region: Ib: 0.0 IIy: 0.0 urx: 0.0 ury: 0.0 ◆ Apply All To Global Net: vss ☐ Override prior connection ☐ Verbose Output Add to List Update Delet	te
<u>Apply</u>	<u>R</u> eset <u>C</u> lose <u>H</u>	elp

hen, we go to the path: **Power → Power Planning → Add Ring**.

Power rings are usually implemented using **intermediate metal layers**, as they need to **reach all areas** of the chip. We make the following changes as shown in the figure below.

	Add Rings -	
asic Ar	Ivanced Via Generation	
Net(s):	vss vdd	
-Ring Typ	e	
 Core 	ring(s) contouring	
🔶 AI	round core boundary 🗼 Along I/O boundary	
🗆 Ex	clude selected objects	
Block	ring(s) around	
- 🔶 E:	ach block	
🔶 E8	ach reef	
🔷 Se	elected power domain/fences/reefs	
🔶 Es	ach selected block and/or group of core rows	
🔶 CI	usters of selected blocks and/or groups of core rows	
-	With shared ring edges	
🔶 User i	defined coordinates: MouseClick	
	Core ring 🕹 Block ring	
- Ring Co	nfiguration	
	Top: Bottom: Left: Right:	
Layer:	METALTH - METALTH - METALZ V - METALZ V -	
width:	0.44 0.44 0.44	1
Spacing:	0.46 0.46 0.46 Update	
Offset:	Center in channel	
	0.56 0.56 0.56	
Ontion (2at	
- opuon a		
_ Use o	otion set: Update Basic	
		-
OK L	Variables Anniu Defaults Cancel Ha	In
<u> </u>		٣

In the figure below, it can be seen that our power rings have been added.



By zooming in, the connections can also be observed.



Next, we need to **add strips** to the circuit. This means adding another layer of metal, which will be connected to **VDD and GND**.

If there is a cell located in the middle of the core and we want to connect it to VDD or GND, it must connect through these **strips**. This is illustrated in the figure below.

Navigate to: **Power** → **Power Planning** → **Add Strip**



Next, we perform a special route. The output will look like the figure shown below.



In the next step, the **final routing** and **metal filler insertion** are performed to complete the layout of the design.

After placing the components, it's time for **routing**.

Before routing the **clock signal (clk)**, we need to add certain cells to the circuit to **prevent clock skew**, **voltage drop on the clock**, and other related issues.

×	
<u>Design Edit Synthesis Partition Floorplan Power Place Clock Route</u>	<u>T</u> iming <u>S</u> I <u>V</u> erify Too <u>l</u> s <u>H</u> elp
Generate Clock Spec	
Specify Buffer/Inverter	
Cells List Selected Cells	
DLY3X1 Add BUFX2	
DLY4X1 CLKBUFX1	
INVX12 Delete	
Output Specification File Cleak attack	
OK Apply Clear Spec Cancel Help	
	A A A A A A A A A A A A A A A A A A A
	A THE SECOND RULE RULE RULE RULE RESOLUTION RESOLUTI RESOLUTION RESOLUTION RESOLUTION RESOLUTION RESOLUTIAN RE

After that, the clk, VDD, and VSS signals are connected to the cells.



We can view the **clock signal routing path** through this window:



Now, we analyze the setup time and hold time for the clk signal. There should be no negative timing values in the output.

Setup mode	I	all	I	reg2reg	I	in2reg	I	reg2out	I	in2out	I	clkgat
WNS	(ns):	14.392		14.798	1	14.392	+-	N/A	1	N/A	-+-	N/A
TNS	(ns):	0.000	I	0.000	I	0.000	T	N/A	I	N/A	I	N/A
Violating Pa	aths:	0	I	0	I	0	I	N/A	I	N/A	I	N/A
A11 P	aths:	534	T	512	I	422	T	N/A	I	N/A	I	N/A
	Nr n	ets(term	ıs)	Worst	Vi	o Nr r	net	s(terms)				
max_cap	1	2 (2)		_0.0	43	1	2	(2)				
max_tran max_fanout		0 (0) 0 (0)		0.00 0	0	- I	0	(0) (0)				
sity: 63.303% ting Overflow orted timing al CPU time:	: 0.009 to dir 1.39 se	6 H and timingF	1. Rep	21% V 								

As shown in the image above, there are no negative values for the setup time. Now, let's examine the hold time:

			SoC
Timing Analysis	<u>S</u> I ⊻erify	Too <u>i</u> s	<u>H</u> elp
Basic Advanced	🔗 🔳	8	<u> </u>
Use Existing Extraction and Timing Data			
<mark>- Design Stage</mark> ↓ Pre-Place ↓ Pre-CTS ◆ Post-CTS ↓ Post-Route ↓ Sign-Off			
Analysis Type ✓ Setup ◆ Hold Include SI			
Reporting Options Number of Paths: 50 Report file(s) Prefix: \$15850 postCTS			
Output Directory: timingReports			
QK Apply Cancel Help			
			The second second second

In the output, we will see that negative values are present, which means the hold time is violated.

Hold mode	all	reg2reg	in2reg	reg2out	in2out	+ clkgate
WNS (ns):	-0.817	0.224	-0.817	N/A	N/A	+ N/A
TNS (ns):	-118.806	0.000	-118.806	N/A	N/A	N/A
Violating Paths:	297	0	297	N/A	N/A	N/A
All Paths:	534	512	422	N/A	N/A	N/A
1131Cy. 05.505/0	0% H and 1	21% V				

In this case, we need to use optimization techniques to try to improve the hold time.



We repeat this process until the hold time becomes positive.

After that, we proceed with routing the remaining signals in the circuit until we reach the layout shown below.



A closer view of the routing layout is shown below:



In the next step, we move on to verifying geometry and connectivity, with the positive results of each step shown below.

Geometry:

*** verity geometry (CPU: 0:00:01.3 MEM: 25.3M)	
encounter 1> *** Starting Verify Geometry (MEM: 367.1) ***	
VERIFY GEOMETRY Starting Verification	
VERIFY GEOMETRY Initializing	
VERIFY GEOMETRY Deleting Existing Violations	
VERIFY GEOMETRY Creating Sub-Areas	
bin size: 8320	
VERIFY GEOMETRY SubArea : 1 of 1	
VERIFY GEOMETRY Cells : O Viols.	
VERIFY GEOMETRY SameNet : O Viols.	
VERIFY GEOMETRY Wiring : O Viols.	
VERIFY GEOMETRY Antenna : O Viols.	
VERIFY GEOMETRY Sub-Area : 1 complete 0 Viols. 0 Wrngs.	
VG: elapsed time: 0.00	
Begin Summary	
Cells : O	
SameNet : O	
Wiring : O	
Antenna : 0	
Short : 0	
Overlap : O	
End Summary	
Verification Complete : O Viols. O Wrngs.	
********End: VERIFY GEOMETRY********	
*** verify geometry (CPU: 0:00:00.0 MEM: 0.0M)	
encounter 1>	
📀 🖬 📀 💽 📜 🛷 💷 🔳	

Connectivity:

***********End: VERIFY GEOMETRY*********
*** verify geometry (CPU: 0:00:01.3 MEM: 25.3M)
encounter 1> *** Starting Verify Geometry (MEM: 367.1) ***
VERIFY GEOMETRY Starting Verification VERIFY GEOMETRY Initializing VERIFY GEOMETRY Deleting Existing Violations VERIFY GEOMETRY Creating Sub-Areas bin size: 8320 VERIFY GEOMETRY SubArea: 1 of 1 VERIFY GEOMETRY SameNet : 0 Viols. VERIFY GEOMETRY SameNet : 0 Viols. VERIFY GEOMETRY SameNet : 0 Viols.
VERIFI GEOMETRY WIFING : 0 Viols.
VERIFY GEOMETRY Sub-Area : 1 complete O Viols. O Wrngs.
VG: elapsed time: 0.00
Begin Summary
Cells : O
SameNet : O
Wiring : O
Antenna : 0
Short : 0
Overlap : O
End Summary
Verification Complete : O Viols. O Wrngs.
*********End: VERIFY GEOMETRY********** *** verify geometry (CPU: 0:00:00.0 MEM: 0.0M)
encounter 1> ******** Start: VERIFY CONNECTIVITY ******* Start Time: Wed Jan 27 10:44:12 2016
Design Name: s15850 Database Units: 2000

After this step, we proceed to **Add Filler**, which fills the gaps between cells to make the layout **manufacturable by the fabrication plant**. After performing this step, the **gaps between cells are filled** as shown below:



Now, we can perform the **metal fill** step: In this step, **all metal layers** are selected, and the **empty spaces** are filled with the corresponding metal. We then arrive at the layout shown below:



And then we **save** the design. **Note:** After several steps, the saved files are available and have been attached. Once the layout is finalized, it must undergo **timing analysis** to verify the correctness of the placement.

In the next step, we **remove the antenna effect** using the **ANTENNATF diode cell**, and then we re-check the **timing analysis**. We observe that there are **no negative values** for either **hold** or **setup** time.

timeDes	ign S	ummary								
Setup mode	+	all	reg2reg	+	2reg	reg2out	-+	in2out	clk	gate
WNS (TNS (Violating Pa All Pa	WNS (ns): 13.831 14.479 13.8 TNS (ns): 0.000 0.000 0.0 olating Paths: 0 0 0 All Paths: 534 512 42		.831 .000 0 422	N/A N/A N/A N/A	 	N/A N/A N/A N/A	N N N	I/A I/A I/A I/A		
DRVs	+ +	Re Nets(terms)	eal +) Worst V	et Vio		Total 	+			
max_cap max_tran max_fanout	 2	2 (2) 2 (122) 0 (0)	-0.08 -0.25 0	3 5	2	2 (2) (122) 0 (0)	+			
sity: 66.886% prted timing t al CPU time: 1	+	timingReg	-+ ports				+			
al Real time: al Memory Usag ounter 1>	1.0 s e: 30	sec 07.417969 1	Mbytes							
🍠 🛛					X			2		

Of course, to make even the **worst violation (worst vio)** positive, we could perform **optimization** multiple times—but this step was skipped.

Another important output of the placement design is the **RC Extraction**, which generates a **.spef** file. This file contains the **parasitic characteristics** of the circuit. We extracted this file from the menu: **Timing** \rightarrow **Extract RC**, and it has been **attached**. Other files such as the **GDS file** and the **netlist** were also extracted and attached.

Final Step: We evaluate, through a **trial-and-error approach**, the **smallest possible Die size** suitable for the design. In this stage, various sizes were tested from the beginning, and the best approximate **Die size** for this circuit was determined to be **around 390×390**.