

OASIS 3D-Router Hardware Physical Design

Technical Report

© Adaptive Systems Laboratory

Division of Computer Engineering School of Computer Science and Engineering University of Aizu

Contact: [d8141104, benab] @ u-aizu.ac.jp

Editions: June 16, 2014, June 22, 2014, July 8, 2014



- 1. Synthesis
- 2. Place & Route
- 3. <u>Design Checking: LVS (Layout-Versus-</u> <u>Schematic) and DRC (Design-Rule Check)</u>
- 4. Post Layout Simulation
- 5. Pad Insertion
- 6. Acknowledgement



<== Back to Contents

1. Synthesis



Objectives

- After completing this tutorial you will be able to:
 - Synthesize 3D-OASIS-NoC (3D-ONoC) router using Design-Compiler CAD tool.
 - Place & Route (P&R) 3D-ONoC router with Cadence SoC-Encounter
 - Evaluate the area and power
 - Learn how to make the synthesis and P&R via:
 - The CAD Graphic User Interface
 - Tcl script



Contents

- Tutorial directory structure
- 3D-OASIS-NoC router hierarchy
- Environment
- Phase1:
 - Design Compiler Synthesis steps (Step 1~7)
- Phase2:
 - SoC Encounter Place & route steps (Step1~12)
- Script

Tutorial directory structure





Tutorial directory structure

Zxp035@zxp035:~	
<u>F</u> ile <u>E</u> dit <u>V</u> iew <u>T</u> erminal Ta <u>b</u> s <u>H</u> elp	
<pre>[zxp035@zxp035 ~]\$ tree -d 3D_ONoC/ 3D_ONoC/ P&R checkpoints input_files reports ' scripts ' Synthesis checkpoints checkpoints output_files reports script ' verilog_src 11 directories [zxp035@zxp035 ~]\$</pre>	

You can check the complete tutorial's directory structure by typing: tree –d 3D_ONoC under your home directory "~"

3D-NoC router hierarchy





Phase 1: Synthesis with Synopsis Design Compiler



Requirements

- In this first phase, we synthesize 3D-ONoC router and evaluate its area and power
- For this phase, we need the Verilog source files which are located in: ~/3D-ONoC/Synthesis/verilog_src
- We also need the .db, .sdb, and .sldb library files which are located in: ~/lib



Synthesis directory structure





Synthesis directory structure

zxp035@zxp035:~/3D_ONoC	
<u>F</u> ile <u>E</u> dit <u>V</u> iew <u>T</u> erminal Ta <u>b</u> s <u>H</u> elp	
<pre>[zxp035@zxp035 3D_ONoC]\$ tree Synthesis/ Synthesis/ Read_me.txt checkpoints output_files reports</pre>	
<pre> script ` syn_LAXYZ.tcl ` verilog_src crossbar.v defines.v fifo.v fifo.v input_port.v matrix_arb_formultistage.v mux_out.v request_cntrl.v route.v route.v route.v stop_go.v</pre>	=
5 directories, 13 files [zxp035@zxp035 3D_ONoC]\$	

You can check the complete Synthesis directory and file structure by typing: 12 tree Synthesis under the "3D_ONoC" directory



Environment

				zxp035@zxp035:~	
<u>F</u> ile <u>E</u> dit	<u>V</u> iew	<u>T</u> erminal	Ta <u>b</u> s	Help	
[zxp035@	zxp035	~]\$ tcsh			
					=

Initially bash will start, so type "tcsh" to start cshr



Environment

	zxp035@zxp035:~	
<u>F</u> ile <u>E</u> dit	t <u>V</u> iew <u>T</u> erminal Ta <u>b</u> s <u>H</u> elp	
<u>File Edit</u> [zxp035@; /home/zxp /home/zxp /home/zxp checkpoin /home/zxp	t <u>V</u> iew <u>lerminal labs H</u> elp 2xp035 ~]\$ tcsh :p035/3D_ONoC% cd Synthesis/ :p035/3D_ONoC/Synthesis% ls .nts output_files Read_me.txt reports script verilog :p035/3D_ONoC/Synthesis% ■	g_src

Go to /home/zxp035/3D-ONoC/Synthesis where the Synthesis folder is located



Environment

Zxp035@zxp035:~	
<u>F</u> ile <u>E</u> dit <u>V</u> iew <u>T</u> erminal Ta <u>b</u> s <u>H</u> elp	
<pre>Ele Edit View Terminal Tabs Help [zxp035@zxp035 ~]\$ tcsh /home/zxp035% cd 3D_ONoC/ /home/zxp035/3D_ONoC% cd Synthesis/ /home/zxp035/3D_ONoC/Synthesis% ls checkpoints output_files Read_me.txt reports script verilog_src /home/zxp035/3D_ONoC/Synthesis% design_vision</pre>	
	*

Type *design_vision* to start Design Compiler



Design Compiler: Synthesis steps

😣 🖨 🗊 🛛 Desi	sign Vision - TopLevel.1	
<u>File E</u> dit <u>V</u> iew	w <u>S</u> elect <u>H</u> ighlight List <u>H</u> ierarchy <u>D</u> esign <u>A</u> ttributes S <u>c</u> hematic <u>T</u> iming <u>T</u> est <u>P</u> ower <u>W</u> indow Help	
] 🚅 🖬 🖨 🖻		
Logical Hie		
	History	Options: 💌
design_v		
Ready		III /



Step 1: Library setup

😣 🖨 🗊 Desig	a a B Design Vision - TopLevel.1					
<u>File E</u> dit <u>V</u> iew	e <u>Edit V</u> iew <u>S</u> elect <u>H</u> ighlight List <u>H</u> ierarchy <u>D</u> esign <u>A</u> ttributes S <u>c</u> hematic <u>T</u> iming <u>T</u> est <u>P</u> ower <u>W</u> indow Help					
🛛 😂 🖓 🖓						
Hier1 Logical Hier						
dc_she design	<pre>L> qu1_start vision></pre>					
Log Hi	tory	Options: 💌				
design_vi						
Ready						

First, we should specify the library for the design. Click file-> Setup 17



Step 1: Library setup a- Target library

😣 💿 Application Setup

Categories	Defaults	
Defaults Variables	Search path: 08.09/libraries/syn /eda/synopsys/syn_vB-2008.09/dw/syn_ver /eda/synopsys/syn_vB-2008.09/dw/sim_ver	
	Physical library:	
	L <u>i</u> nk library: * * your_library.db	
	Target library: * your_library.db	
	Symbol library: * your_library.sdb	
	Synt <u>h</u> etic library:	
	* - required	
I		
	<u>R</u> eset ▼ OK Cancel <u>A</u> pply	/



Step 1: Library setup a- Target library

😣 🗈 Set Target Libraries		
Target library: *		
your_library.db		<u>A</u> dd
		<u>D</u> elete
		Û
		Û
	ок	Cancel

Click on you_library.db to modify the default library



Step 1: Library setup a- Target library

😣 🗊 Select Files	
Look in: 🔄 /home/zxp035/lib/ 🔽 🗲 🔁	d* 🔝 🎟
🗀 🗋 iopad.lef	
📄 Back_End 📑 slow.db	
📄 Front_End 📑 slow.lib	
📄 Low_Power 📑 typical.db	
🕞 virtuoso 🔄 🕞 typical.lib	
LICENSE	
README	
ि cells.lef	
ੁਰੇ cells.v	
ਜ਼੍ਹੇ fast.db	
🕞 fast.lib	
File <u>n</u> ame: "typical.db"	<u>O</u> pen
File type: All files (*)	Cancel



Step 1: Library setup b- Symbol library

😣 🗊 Application Setup				
Categories	Defaults			
Defaults Variables	Search path: 08.09/libraries/syn /eda/synopsys/syn_vB-2008.09/dw/syn_ver /eda/synopsys/syn_vB-2008.09/dw/sim_ver			
	Physical library:			
	Link library: * * your_library.db			
	Target library: * /home/zxp035/lib/typical.db			
	Symbol library: * your_library.sdb			
	Synt <u>h</u> etic library:			
	* = required			
	Reset OK Cancel Apply			



Step 1: Library setup b- Symbol library

Application Setup	
Categories Defaults	
Default: 8 Select File	n ver (eda/synonsys/syn vB-2008.09/dw/sim ver
Variables Look in: 🔄 /home/zxp035/lib/	
Image: Section of the section of th	
File name: generic.sdb	OK Cancel
File type: All files (*)	Reset - OK Cancel Apply



Step 1: Library setup c- Synthetic library

😣 🔲 Application Setup

Categories	Defaults
Defaults Variables	Search path: 08.09/libraries/syn /eda/synopsys/syn_vB-2008.09/dw/syn_ver /eda/synopsys/syn_vB-2008.09/dw/sim_ver
	Physical library:
	L <u>i</u> nk library: * * your_library.db
	Target library: * /home/zxp035/lib/typical.db
	Symbol library: * /home/zxp035/lib/generic.sdb
	Synthetic library:
	* = required
	<u>R</u> eset • OK Cancel <u>A</u> pply



Step 1: Library setup c- Synthetic library

🗴 🗉 Application Setup	
Stagering 8 Select Files	
- Default Look in: 🔄 /home/zxp035/lib/ ▼ 🗘 🏠 👬 🏥	vB-2008.09/dw/sim_ver
Variable Image: Construct of the state of the stat	Add Delete ① ① ① ① ① ①
File type: All files (*) Cancel	
	Reset v OK Cancel Apply



Step 1: Library setup d- Link library

8 Application	Setup
Categories	Defaults
- Defaults 	Search path: b8.09/libraries/syn /eda/synopsys/syn_vB-2008.09/dw/syn_ver /eda/synopsys/syn_vB-2008.09/dw/sim_ver Physical library: Link library: * /home/zxp035/lib/typical.db /home/zxp035/lib/dw_foundation.sldb Target library: * /home/zxp035/lib/typical.db Symbol library: * /home/zxp035/lib/typical.db Symbol library: * /home/zxp035/lib/typical.db Symbol library: * /home/zxp035/lib/dw_foundation.sldb
	* = required <u>Reset</u> OK Cancel <u>Apply</u>

Finally the *Link library*, should contain the combined path of:

- Current directory: "*"
- Target library: "home/zxp035/lib/typical.db"
- Synthetic library: "/home/zxp035/lib/dw_foundation.sldb"



Step 2: Analysis a- File selection

😣 🗉 Analyze Designs	
Fi <u>l</u> e names in analysis order:	
	A <u>d</u> d
	D <u>e</u> lete
	*
Format: Auto	~
Work library: WORK	v
Create new library if it does no	t exist
ок	Cancel

After setting the libraries, we should make the analysis of our circuit Click on File->Analyze



Step 2: Analysis a- File selection

😣 🗈 Ana	lyze Designs		🙁 🗐 An	alyze Designs	
Fi <u>l</u> e names in	analysis order:		Fi <u>l</u> e names i	in analysis order:	
		A <u>d</u> d			A <u>d</u> d
		D <u>e</u> lete			D <u>e</u> lete
		*			*
<u>F</u> ormat:	VERILOG (v)	•	<u>F</u> ormat:	VERILOG (v)	-
<u>W</u> ork library:	Auto		Work library	WORK	7
	VERILOG (V) SVERILOG (sv) VHDL (vhdl)	Cancer		Create new library if it does no	ot exist Cancel
	1			2	



Step 2: Analysis a- File selection

😣 🗊 Analyze Designs	
Look in: 🔄 /home/zxp035/3D_ONoC/Synthesis/verilog_src/ 💽 存 🔁	🖄 🔠 🏛
crossbar.v	
input_port.v	
matrix_arb_formultistage.v mux_out.v	
request_cntrl.v	
router_LAXYZ.v	
stop_go.v	
File <u>n</u> ame: st_cntrl.v" "route.v" "router_LAXYZ.v" "stop_go.v" "sw_alloc.v"	<u>S</u> elect
File type: Database Files (*.v *.vhd *.sv *.vhdl)	Cancel

Go to *./verilog_src* and select the 11 Verilog files of the router 28 Click Select



Step 2: Analysis a- File selection

😣 🗈 🛛 Analyze Designs

File names in	analysis order:		
/home/zxp0	35/3D_ONoC/Synthesis/verilog_src/sw_alloc.v		
/home/zxp0	35/3D_ONoC/Synthesis/verilog_src/stop_go.v		A <u>d</u> d
/home/zxp0	35/3D_ONoC/Synthesis/verilog_src/router_LAXYZ.v		3
/home/zxp03	35/3D_ONoC/Synthesis/verilog_src/route.v		
/home/zxp0	35/3D_ONoC/Synthesis/verilog_src/request_cntrl.v		D <u>e</u> lete
/home/zxp0	35/3D_ONoC/Synthesis/verilog_src/mux_out.v		
/home/zxp0	35/3D_ONoC/Synthesis/verilog_src/matrix_arb_formultista	ge.v	
/home/zxp03	35/3D_ONoC/Synthesis/verilog_src/input_port.v		
/home/zxp03	35/3D_ONoC/Synthesis/verilog_src/fifo.v		1
/home/zxp0	35/3D_ONoC/Synthesis/verilog_src/defines.v		
/home/zxp0	35/3D_ONoC/Synthesis/verilog_src/crossbar.v		
<u>F</u> ormat:	VERILOG (v)		•
<u>W</u> ork library:	WORK		V
	Create new library if it does not exist		
	[ОК	Cancel



Rea

Step 2: Analysis b- Analysis report

Searching for /eda/synopsys/syn_vB-2008.09/dw/sim_ver/defines.v	
Searching for /home/zxp035/3D_ONoC/Synthesis/verilog_src/defines.v	
Opening include file /home/zxp035/3D_ONoC/Synthesis/verilog_src/defines.v	
Compiling source file /home/zxp035/3D_ONoC/Synthesis/verilog_src/fifo.v	
Searching for ./defines.v	
Searching for /eda/synopsys/syn_vB-2008.09/libraries/syn/defines.v	
Searching for /eda/synopsys/syn_vB-2008.09/dw/syn_ver/defines.v	
Searching for /eda/synopsys/syn_vB-2008.09/dw/sim_ver/defines.v	
Searching for /home/zxp035/3D_ONoC/Synthesis/verilog_src/defines.v	
Opening include file /home/zxp035/3D_ONoC/Synthesis/verilog_src/defines.v	
Compiling source file /home/zxp035/3D_ONoC/Synthesis/verilog_src/defines.v	
Compiling source file /home/zxp035/3D_ONoC/Synthesis/verilog_src/crossbar.v	
Presto compilation completed successfully.	
design_vision>	
og Uiston	
History	
esign vision>	_
L d	Searching for /eda/synopsys/syn_vB-2008.09/dw/sim_ver/defines.v Searching for /home/zxp035/3D_ONoC/Synthesis/verilog_src/defines.v Opening include file /home/zxp035/3D_ONoC/Synthesis/verilog_src/fifo.v Searching for ./defines.v Searching for /eda/synopsys/syn_vB-2008.09/libraries/syn/defines.v Searching for /eda/synopsys/syn_vB-2008.09/dw/syn_ver/defines.v Searching for /eda/synopsys/syn_vB-2008.09/dw/sim_ver/defines.v Searching for /eda/synopsys/syn_vB-2008.09/dw/sim_ver/defines.v Searching for /eda/synopsys/syn_vB-2008.09/dw/sim_ver/defines.v Searching for /eda/synopsys/syn_vB-2008.09/dw/sim_ver/defines.v Searching for /home/zxp035/3D_ONoC/Synthesis/verilog_src/defines.v Opening include file /home/zxp035/3D_ONoC/Synthesis/verilog_src/defines.v Compiling source file /home/zxp035/3D_ONoC/Synthesis/verilog_src/defines.v Compiling source file /home/zxp035/3D_ONoC/Synthesis/verilog_src/crossbar.v Presto compilation completed successfully. design_vision>



Step 2: Analysis c- Checkpoint

😣 🗉 Save Design As	
Look in: 🔄 /home/zxp035/3D_ONoC/Synthesis/checkpoints/	
, File <u>n</u> ame: analysis ddc	<u>S</u> ave
File type: Database Files (*.ddc *.ddc.gz *.db *.db.gz *.gdb *.sdb *.pdb *.eqn *.fn	Cancel
Eormat: DDC (ddc)	Synopsys [,]
✓ Save all designs in hierarchy	

- 1. Click File-> Save As
- 2. Go to ./checkpoints
- 3. In File name, type analysis.ddc
- 4. Change the Format to DDC (ddc)
- 5. Click Save

It is recommended to save your progress after each step



Step 3: Elaboration a- Top file selection

😕 🗊 Elaborate Designs		
<u>L</u> ibrar y :	DEFAULT	
<u>D</u> esign:	router_LAXYZ(verilog)	
<u>P</u> arameters:	crossbar(verilog) fifo(verilog)	
Reanalyze	input_port(verilog) matrix_arb_formultistage(verilog) mux_out(verilog) request_cntrl(verilog) route(verilog)	
<u>r R</u> eanaly26	router_LAXYZ(verilog) stop_go(verilog) sw_alloc(verilog)	

After Analysis, we should make the Elaboration of our circuit. Click on File->Elaborate In Design, select the top module of the router router_LAXYZ (verilog)



Step 3: Elaboration a- Top file selection

😣 🗊 🛛 Elat	porate Designs	5	
<u>L</u> ibrary:	DEFAULT	•	
<u>D</u> esign:	router_LAXYZ(verilog)		
<u>P</u> arameters:	Name	Value	
∏ <u>R</u> eanalyze	e out-of-date libr	aries	
		OK Cancel	



Step 3: Elaboration b- Elaboration report

	Information: Building the design 'mux_out' instantiated from
	the parameters "7,34". (HDL-193)
	Warning: Starting with the 2000.11-1 release, the Presto Ve
	Presto compilation completed successfully.
	design_vision>
	Current design is 'router_LAXYZ'.
L	.og History
de	esign_vision>
leady	



Step 3: Elaboration c- Hierarchy

Design Vision - TopLevel.1 (router_LAXYZ)	
<u>File Edit View Select Highlight List H</u> ierarchy <u>D</u> esign <u>A</u> ttributes	S <u>c</u> hematic <u>T</u> iming <u>T</u> est <u>P</u> ower <u>W</u> indov
🍰 🔲 🕼 👔 🔍 Q. Q. 🔍 Q. 🕼 🛛 🖆 🔝 🔝 🔜 🔜	🗃 🔜 🔛 📰 📗 router_LAXYZ
Relier.1	
Logical Hierarchy Cells (Hierarchical)	
(i) G==> router_LAXYZ Cell Name Ref Name Cell	Path
(€) ([0].ip ([0].ip input_port_N il[0].i	p
⊡ @ il[1].ip @ il[1].ip input_port_N il[1].i	p
⊕ @ il[2].ip @ il[2].ip input_port_N il[2].i	p
⊡ @il[3].ip @il[3].ip input_port_N il[3].i	p
© il[4].ip © il[4].ip input_port_N il[4].i	p
⊡ ©il[5].ip @il[5].ip input_port_N il[5].i	p
⊡ ©il[6].ip @il[6].ip input_port_N il[6].i	p
⊡ © sw_allc	ilic
🗄 🕼 cbar 🕼 cbar crossbar_NO cbar	

The design hierarchy can be seen on the left side of the window



Step 3: Elaboration d- Schematic



The design schematic can be seen by clicking on this icon **D** at the top of the window


Step 3: Elaboration e- Checkpoint

😣 🗉 Save Design As	
Look in: 🔄 /home/zxp035/3D_ONoC/Synthesis/checkpoints/ 💽 🗢 🗄	
File <u>n</u> ame: elaboration.ddc	<u>S</u> ave
File type: Database Files (*.ddc *.ddc.gz *.db *.db.gz *.gdb *.sdb *.pdb *.eqn *.fn 💌	Cancel
Format: DDC (ddc)	SYNOPSYS'
Save all designs in hierarchy	

- 1. Click File-> Save As
- 2. Go to ./checkpoints
- 3. In File name, type elaboration.ddc
- 4. Make sure the Format is DDC (ddc)
- 5. Click Save

Step 4: Constraints setting a- Clock

- 1. Set clock name :
 clk

 2. Set Period:
 10.0
- 2. Set Period. 10.0
- 3. Set *Rising*: **0.00**
- 4. Set *Falling*: **5.00**
- 5. Check Don't touch network

Click OK

•	🕽 🗊 Specify C	lock		
<u>c</u>	lock name: clk	1		
P	ort name:			
	<u>R</u> emove clock			
	Clock creation –			
	Perio <u>d</u> : 10.0	2		
2	Edge	Value		Add <u>e</u> dge pair
	Rising Falling		0.00 5.00	Rem <u>o</u> ve edge pair
4		•		Invert <u>w</u> ave form
	ļ			
			7	Γ
	0.00	_	5.00	10.
	Don't <u>t</u> ouch ne	twork D		Fix <u>h</u> old
		ОК		Cancel <u>A</u> pply

Click Attributes -> Specify Clock

Step 4: Constraints setting b- Wire load

😣 🗊 🛛 Wire Load	ł
Current design:	router_LAXYZ
Wire load model:	5K_hvratio_1_1 (NangateOpenCellLibrary)
5K_hvratio_1_1	(NangateOpenCellLibrary)
5K_hvratio_1_2	(NangateOpenCellLibrary)
5K_hvratio_1_4	(NangateOpenCellLibrary)
3K_hvratio_1_1	(NangateOpenCellLibrary)
3K_hvratio_1_2	(NangateOpenCellLibrary)
	OK Cancel <u>A</u> pply

- Click Attributes -> Operating Environment-> Wire Load...

- Select 5K_hvratio_1_1 and then click OK



Step 5: Compilation

- 1. Set Map effort to high
- 2. Check Incremental mapping

Click OK

😣 🗈 Compile	
Mapping options	Compile options
<mark>™</mark> <u>M</u> ap design	🗖 Top leve 🔽 Incremental mapping
🔽 Exact map	「 Ungroup 「 Allo <u>w</u> boundary conditic
Ma <u>p</u> effort: high	「 <u>S</u> can 「 Auto <u>u</u> ngroup
Ar <u>e</u> a effort: medium 🗨	Cate Cla
Po <u>w</u> er effort: medium 💌	C Delay
Design rule options	
• Fix <u>d</u> esign rules and optimize map	ping
C Optimize mapping onl <u>y</u>	
C Fix design ru <u>l</u> es only	
C Fix hold time only	
	OK Cancel <u>A</u> pply



Step 6: Report a- Area

😣 🗉 Report Area
-Report for-
Current design: router_LAXYZ Current instance:
-Report options
☑ No <u>l</u> ine split
-Output options
☑ To report <u>v</u> iewer
☐ To <u>f</u>ile : Report.txt <u>B</u> rowse
✓ Append to file
OK Cancel <u>A</u> pply



Step 6: Report a- Area

🗒 Report.1 - Area		
XP		M
*****	*****	* *
Report : area		
Design : router_LAXYZ		
Version: B-2008.09		
Date : Mon Jun 2 00:23	3:31 2014	
*****	****	**
Library(s) Used:		
NangateOpenCellLibrar	ry (File: /home	e/zxp035/lib/typical.db)
Number of ports:	501	
Number of nets:	865	
Number of cells:	23	
Number of references:	10	
Combinational area:	5522.692051	
Noncombinational area:	4926.851833	
Net Interconnect area:	undefined	(Wire load has zero net area)
matel	10440 543004	
Total cell area:	10449.543884	
Total area:	underined	

In this library the wire load models do not include area information (Wire load has zero net area) so the Net Interconnect area (and therefore Total area) is left undefined. The area is measured in micrometer (um)



Step 6: Report b- Power

😣 🗉 Report Power	
-Report for	
Summary only 🔻	
All nets/cells [<u>q]</u> Onl <u>y</u> nets/cells:	<u>Selection</u>
Report options	
Show nets histogram Exclude values <= Exclude values >=	Use hierarchical format[<u>z]</u> Hierarchy levels:
Analysis effo <u>r</u> t: low	Sort <u>m</u> ode:
☐ No <u>l</u> ine split	☐ Verb <u>o</u> se
Exclude power of boundary nets	Report cumulative power[k]
Traverse hierarchy at all levels	
- Output options	
☑ To report <u>v</u> iewer	
To file: Report.txt	Browse
Append to file	
	OK Cancel <u>A</u> pply

Click on **Design-> Report Power** then **OK**.



Step 6: Report b- Power

Report 3 - Power	IX
X 2 A	
Report : power	
-analysis_effort low	
Design : router_LAXYZ	
Version: B-2008.09	
Date : Mon Jun 2 00:52:59 2014	
Library(s) Used:	
NangateOpenCellLibrary (File: /home/zxp035/lib/typical.db)	
Operating Conditions: typical Library: NangateOpenCellLibrary	
Wire Load Model Mode: top	
Design Wire Load Model Library	
router_LAXYZ 5K_nvratio_1_1 NangateOpenceIlLibrary	
Global Operating Voltage = 1.1	
Power-specific unit information :	
Voltage Units = 1V	
Capacitance Units = 1.000000ff	
Time Units = 1ns	
Dynamic Power Units = 1uW (derived from V,C,T units)	
Leakage Power Units = 1nW	
Cell Internal Power = 704.9396 uW (84%)	
Net Switching Power = 136.0538 uW (16%)	
Total Dynamic Power = 840.9933 uW (100%)	
Call Laskage Dever = 201 2620 NM	
Cell Leakage Power = 201.2622 UW	-

The report shown above should appear giving information about Cell Internal Power, Net Switching Power, Total Dynamic Power, and the Cell Leakage Power, in addition to their percentage from the total power consumption.



Step 7: Output files a- Verilog file

😣 🗈 Save Design As	
Look in: 🔄 /home/zxp035/3D_ONoC/Synthesis/output_files/] 💣 🔝 🏛
File <u>n</u> ame: router_LAXYZ.vnet	<u>S</u> ave
File type: Database Files (*.ddc *.ddc.gz *.db *.db.gz *.gdb *.sdb *.pdb *.eqn *.fn 💌	Cancel
Format: VERILOG (v)	SYNOPSYS'
☑ <u>S</u> ave all designs in hierarchy	

- 1. Click File-> Save As
- 2. Go to ./output_files
- 3. In *File name*, type router_LAXYZ.vnet
- 4. Change the Format to Verilog (V)
- 5. Click Save

After finishing the compilation, we should generate the necessary files for the next 45 Place&Route step



Step 7: Output files b- sdc file

Warning: Design 'router_LAXYZ' contains 1 high-fanout nets. A fanout number of 1000 will be used for delay calculations involving these nets. (TIM-134)
Net 'sw_allc/ol[2].spg/clk': 1071 load(s), 1 driver(s)	
Writing verilog file '/home/zxp035/Desktop/3D-NoC/LAXYZ/output_files/router_LAXYZ.vnet'.	
Warning: Verilog 'assign' or 'tran' statements are written out. (VO-4)	
Writing ddc file './DB/router_LAXYZ.ddc'.	
design_vision>	
Current design is 'router_LAXYZ'.	
Log History	
design_vision> write_sdc ./output_files/router_LAXYZ.sdc	

In the dc_shell window type the following command to save the .sdc file write_sdc ./output_files/router_LAXYZ.sdc



Step 7: Output files e- checkpoint

😣 🗉 Save Design As	
Look in: 🔄 /home/zxp035/3D_ONoC/Synthesis/checkpoints/	• 🗗 🏗
File <u>n</u> ame: router_LAXYZ.ddc	<u>S</u> ave
File type: Database Files (*.ddc *.ddc.gz *.db *.db.gz *.gdb *.sdb *.pdb *.eqn *.fn	Cancel
Eormat: DDC (ddc)	SYNOPSYS'
✓ Save all designs in hierarchy	

- 1. Click File-> Save As
- 2. Go to ./checkpoints
- 3. In File name, type router_LAXYZ.ddc
- 4. Make sure the Format is DDC (ddc)
- 5. Click Save



Scripts

- The 7 steps previously presented can be made via commands inserted in the dc_shell.
- The commands required for the synthesis are grouped in a single .*tcl* file.
- The .tcl file is named "syn_LAXYZ.tcl"
- It is located in:
 "/home/zxp035/3D_ONoC/Synthesis/script



Script

😣 🗈 Execute Script File	
Look in: 🔄 /home/zxp035/Desktop/3D-NoC/LAXYZ/scripts/	• 🗈 💣 🔡 🏢
svn LAXYZ tcl	
File <u>n</u> ame: syn_LAXYZ.tcl	<u>O</u> pen
File type: Script Files (*.script *.scr *.dcs *.dcv *.dc *.dcfpga *.con *.tcl *.tcl)	- Cancel
Echo commands	
✓ Verbose	

To run the TCL script, click **File> Execute script** Go to ./*scripts, select syn_LAXYZ.tcl* and click *Open*



Script: syn_LAFT.tcl (1/3)

Define the variable which we will use
set base_name "router_LAXYZ"
set clock_name "clk"
set clock_period 10.0
Step 1: Set the libraries:

set target_library "~/lib/typical.db"
set synthetic_library "~/lib/dw_foundation.sldb"
set link_library [concat "*" \$target_library \$synthetic_library]
set symbol_library ""~/lib/generic.sdb"
define_design_lib WORK -path ./WORK # redirect the log files to a new folder "WORK"

Step 2: Analysis

analyze -format verilog {./verilog_src/crossbar.v ./verilog_src/defines.v ./verilog_src/fifo.v ./verilog_src/input_port.v ./verilog_src/matrix_arb_formultistage.v ./verilog_src/mux_out.v ./verilog_src/request_cntrl.v ./verilog_src/route.v ./verilog_src/router_LAXYZ.v ./verilog_src/stop_go.v ./verilog_src/sw_alloc.v}



Script: syn_LAFT.tcl (2/3)

Analysis checkpoint

write_file -format ddc -hierarchy -output ./checkpoints/analysis.ddc

Step 3: Elaboration#### elaborate \$base name # Elaboration checkpoint write file -format ddc -hierarchy -output ./checkpoints/elaboration.ddc #### Step 4: Constraints#### # Clock create clock -name \$clock name -period \$clock period [find port \$clock name] set clock uncertainty 0.02 [get clocks \$clock name] # Delay set_input_delay 0.1 -clock clk [remove_from_collection [all inputs] {clk reset}] set output delay 0.1 -clock clk [all outputs] # Wire load

set_wire_load_model -name 5K_hvratio_1_1 -library NangateOpenCellLibrary



Script: syn_LAFT.tcl (3/3)

Step 5: Compilation####

compile -map_effort high
compile -incremental_mapping -map_effort high

Step 6: Report####
Summary report to be saved under the "reports" folder
report_qor > ./reports/Summary_report_\${base_name}.txt
Hierarchical area report to be saved under the "reports" folder
report_area -hierarchy > ./reports/report_area_\${base_name}.txt

```
#### Step 7: Output files ####
# verilog file
write -format verilog -hierarchy -output ./output_files/${base_name}.vnet
# sdc file
write_sdc ./output_files/${base_name}.sdc
# Final checkpoint
write_file -format ddc -hierarchy -output ./DB/${base_name}.ddc
```



<== Back to Contents

2. Place & Route



Requirements

- After we finished the synthesis phase, we proceed to perform the Place and Route of 3D-ONoC router with Cadence SoC Encounter.
- For this phase, we need the .**vnet** and .**sdc** files obtained from the synthesise phase and which are located in:

~/3D-ONoC/Synthesis/output_files

 We also need the .lib and .lef library files which are located in: ~/lib



Place and Route directory structure





Place and Route directory structure

<pre>Ele Edit View Terminal Tabs Help [zxp035@zxp035 ~]\$ cd 3D_ONoC/ [xp035@zxp035 3D_ONoC]\$ tree P\&R/ P&R/ checkpoints input_files reports ` scripts ` par_LAFT.tcl 4 directories, 1 file [zxp035@zxp035 3D_ONoC]\$</pre>		zxp035@zxp035:~/3D_ONoC	
<pre>[zxp035@zxp035 ~]\$ cd 3D_ONOC/ [zxp035@zxp035 3D_ONOC]\$ tree P\&R/ P&R/ checkpoints input_files reports ` scripts ` par_LAFT.tcl 4 directories, 1 file [zxp035@zxp035 3D_ONoC]\$</pre>	<u>F</u> ile <u>E</u> dit <u>V</u> iew <u>T</u> erminal	Ta <u>b</u> s <u>H</u> elp	
4 directories, 1 file [zxp035@zxp035 3D_ONoC]\$	<pre>[zxp035@zxp035 ~]\$ cd 3[[zxp035@zxp035 3D_0NoC]\$ P&R/ checkpoints input_files reports ` scripts ` par LAFT.tcl</pre>)_ONoC/ ⊱ tree P\&R/	
	4 directories, 1 file [zxp035@zxp035 3D_ONoC]\$	5	

You can check the complete Synthesis directory and file structure by typing: 56 tree Synthesis under the "3D_ONoC" directory



Environment

	zxp035@zxp035:~	
<u>F</u> ile <u>E</u> dit <u>V</u> iew <u>T</u> erminal Ta <u>b</u> s <u>H</u> elp		
[zxp035@zxp035 ~]\$ tcsh		
/home/zxp035% cd 3D_0NoC/ /home/zxp035/3D 0NoC% cd P\&R/		
/home/zxp035/3D_0NoC/P&R%		
		=
		*

Make sure that you are working under **cshr** environment. Otherwise type **tcsh.** Go to /home/zxp035/3D-ONoC/P&R where the P&R folder is located

Environment

zxp035@zxp035:~	
<u>F</u> ile <u>E</u> dit <u>V</u> iew <u>T</u> erminal Ta <u>b</u> s <u>H</u> elp	
<pre>[zxp035@zxp035 ~]\$ tcsh /home/zxp035% cd 3D_ONoC/ /home/zxp035/3D_ONoC% cd P\&R/ /home/zxp035/3D_ONoC/P&R% cp/Synthesis/output_files/router_LAXYZ.vnet ./input_files/ /home/zxp035/3D_ONoC/P&R% cp/Synthesis/output_files/router_LAXYZ.sdc ./input_files/ /home/zxp035/3D_ONoC/P&R%</pre>	
	Ţ

First, we need t copy the *router_LAXYZ.vnet* and *router_LAXYZ.sdc* files generated from the synthesis phase which will be used as input for the P&R phase. Type: % cp ../Synthesis/ouput_files/router_LAXYZ.vnet ./input_files % cp ../Synthesis/ouput_files/router_LAXYZ.sdc ./input_files



Environment

Zxp035@zxp035:~	L_I_IX
<u>F</u> ile <u>E</u> dit <u>V</u> iew <u>T</u> erminal Ta <u>b</u> s <u>H</u> elp	
<pre>[zxp035@zxp035 ~]\$ tcsh /home/zxp035% cd 3D_ONoC/ /home/zxp035/3D_ONoC% cd P\&R/ /home/zxp035/3D_ONoC/P&R% cp/Synthesis/output_files/router_LAXYZ.vnet ./input_files/ /home/zxp035/3D_ONoC/P&R% cp/Synthesis/output_files/router_LAXYZ.sdc ./input_files/ /home/zxp035/3D_ONoC/P&R% velocity</pre>	

Type velocity to start SoC Encounter



SoC Encounter: P&R steps

😣 🖲 🗉 Encounter(R) RTL-to-GDSII System 10.1 - /home/zxp035/3D_ONo(C/P&R -
Eile Edit View Partition Floorplan Power Place Optimize Clock Route Timing Verif	fy Options Tools Flows Help cādence
	All Colors
	Physical Layers
	Physical Layers 🛛 🗹 🗹
	Instance 🔛 🗹 🗹
	Std. Cell 🔤 🗹 🗹
	Physical Cell 🔛 🗹 🗹
	Cover Cell Si 🖉 💆
	₽/G ✓
	Routing Blkg 🛛 🗹 🗹 🚽
	Obstruct 📃 🗹 🗹
	Cell Blockage
	Standard Row
	Metal Fill 🔤 🗹 🗹
	Violation 🔤 🗹 🗸
	Net 📃 💆
	Special Net
	Wire/Via Lavers V
	Ma 01 🛛 🖉 🗹
	Metal 1 🔤 🗹 🗹
	Via 12
	Metal 2
	Metal 3
	Via 34 📃 🗹 🚽
	World View
Click to select single object. Shift+Click to de/select multiple objects.	Q SelNum:0 (0.130, 0.036) Not in Memory

Welcome screen



Step 1: Import Design

🕒 🗈 Design Import	
Basic Advanced	
Netlist:	
 Verilog 	_
Files:	
	Top Cell: 🔾 Auto Assign 🖲 By User:
O OA	
Library:	•
Cell:	•
View:	
Technology/Physical Librarie	25:
LEF Files:	
OA Reference Libraries:	
OA Abstract View Names:	
OA Layout View Names:	
Floorplan	
IO Assignment File:	B
Analysis Configuration	
MMMC View Definition File:	8
	Create Analysis Configuration

Click on **File->Import Design** Click **Files** to import the netlist



Step 1: Import Design a- Netlist (.vnet)

😣 💷 Netlist Files	
Netlist File:	Natlist File: input files /router LAY/Zumet Add Con Netlist Selection:
Netlist Files:	Netlist Files: 2 2 /home/zxp035/3D_ONoC/P&R/input_files
	input_files/router_LAXYZ.vnet
	Filters: Netlist Files (*.v*)
Delete	
Close	

- 1. Click on >> to expand
- 2. Go to ./input_files folder
- 3. Double click on router_LAXYZ.vnet
- 4. Click Close



Step 1: Import Design b- Top module

ic Advanced		
Netlist:		
Verilog		
Files:	input_files/router_LAXYZ.vnet	
	Top Cell: 🔾 Auto Assign 💿 By User: 🛛 router_LAXYZ	
) oa		
Library:		-
Cell:		-
View:		-
Technology/Physical Libr	aries:	
LEF Files:		
OA Reference Libraries:		
OA Abstract View Names:		
OA Layout View Names:		
Floorplan		51
IO Assignment File:		6
Analysis Configuration		
MMMC View Definition File	1	D
	Create Analysis Configuration	

- 1. In Top Cell: type router_LAXYZ
- 2. Click on LEF files



Step 1: Import Design c- LEF file

😣 🗉 LEF Files	
LEF File:	Image: A constraint of the second of the
Close	Close 4

- 1. Click on >> to expand
- 2. Go to ~/lib folder
- 3. Double click on NangateOpenCellLibrary.lef
- 4. Click Close



Step 1: Import Design d- Advanced settings

🛿 😑 🗉 Design Import	😣 🗖 🗊 Design Import
Basic Advanced	Basic Advanced
Netlist:	Timing
• Verilog	2 IPO/CTS CTS Cell List: CLKBUF_X1 CLKBUF_X2 CLKBUF_X3
Files: input_files/router_LAXY2pet	Power BTL
Top Cell: O Auto Assign S By User: router_LAXYZ	Yield
O OA	
Library:	
Cell:	
View:	
Technology/Physical Libraries:	
LEF Files:/ib/NangateOpenCellLibrary.lef	
OA Reference Libraries:	
OA Abstract View Names:	
OA Layout View Names:	
Floorplan	
IO Assignment File:	
Analysis Configuration	
MMMC View Definition File:	
Create Analysis Configuration	
OK Save Load Cancel Help	OK Save Load Cancel Help

- 1. Click on Advanced
- 2. In IPO/CTS type CLKBUF_X1 CLKBUF_X2 CLKBUF_X3



Step 1: Import Design d- Advanced settings

2

- 1. In Power type:
 - a. VDD in Power nets
 - b. VSS in Ground Nets
- Click back on Basic (DO NOT click on OK)

ILM	Fower Mers ADD	
IPO/CTS Power	Ground Nets VSS	
RTL	Toggle Rate Scale Factor:	1.0
Yield		

Invilor	
Verilog Files	innut files/router I AVV7.unet
110.01	TOD Cell: Auto Assim Profilery volutor LAVV7
○ OA	Top com C Allo Assign C by oser. Totale_LAATE
Library:	
Cell:	
View:	
OA Reference Libraries: OA Abstract View Names: OA Layout View Names: Floomlan	
IO Assignment File:	<u>₽</u>
Analysis Configuration	
MMMC View Definition File	

Click on Create Analysis Configuration



- 1. Double click on Library Sets in the MMMC Browser window
- 2. On the add Library Set window, type default in Name
- 3. Click on Add.
- 4. In the Timing Library Window, go to ~/lib and select typical.lib
- 5. Click Open (Timing Library Window) and then OK (Library Set Window)



Add Delay Corner Name: default Wizard Help Power Domain List Type B: Addyris Wews B: - Library Set: B: Occord at the recessary data available, it is recommended that you calculation, and timing analysis. This wizard will assist you in specifying the recessary data available, it is recommended that you calculation, and timing analysis. B: Hold Analysis Wews B: Corners It you have all the recessary data available, it is recommended that you calculation, and timing analysis. B: Hold Analysis Wews B: Condition B: Condition It you have all the recessary data available, it is recommended that you calculation, and timing analysis. B: Hold Analysis Wews B: Condition It you have all the recessary data available, it is recommended that you calculation, and timing analysis. B: Do Condition OpCondition It you have all the recessary data available, it is recommended that you condition and the recessary data available, it is recommended that you condition and the recessary data available, it is recommended that you condition and the received that you condition and the received that you condition and the received the received the received that you condition and the received the received the received that you available. Ibrary Set Ibrary Set Ibrary Set Ibrary Set Ibrary Set Ibrary Set Ibrary Set Ibrary Set Ibrary Set Ibrary Set Ibrary Set Ibrary Set Ibrary Set </th <th></th> <th></th> <th></th> <th></th> <th></th>					
Name:	😣 🚍 🗉 🛛 Add Delay Corner		Analysis View List	MMMC Objects	Wizard Help
	Add Delay Corner Name: default Power Domain List default Add Delere	Type On Chip Variation Single/BcWc Attributes RC Corner: Library Set default Op Cond Lib: Op Cond: IrDrop File: Early Library Set: Op Cond: IrDrop File: Early Library Set: Op Cond: IrDrop File: Concord Lib: Op Cond: IrDrop File: Concord Lib: Op Cond: IrDrop File: Concord Lib: Op Cond: Concord Lib: Op Cond: Concord Lib: Op Cond: Concord Lib: Concord	Analysis View List Analysis Views Setup Analysis Views Hold Analysis Views	MIMC Objects Library Sets Br-default RC Corners Br-OP Conds Drelay Corners Br-Convariant Modes 1	Wizard Help This wizard will assist you in specifying the necessary information to configure the system for RC extraction, delay calculation, and timing analysis. It you have all the necessary data available, it is recommended that you configure the system as completely as possible for all steps of the implementation flow - through signoff. If not, you can always update the configuration, if necessary, as you proceed through the flow. If you are comfortable using the MIMINC Browser, you can use the Wizard Off button to remove the help dialog, and proceed at your own pace. For additional assistance with design import, press the Next button

- 1. Double click on **Delay Corners** in the MMMC browser window
- 2. On the Add Delay Corner window, type default in Name
- 3. Change the *Library Set* to **default**.
- 4. Click OK



- 1. Double click on **Constraint Modes** in the MMMC browser window
- 2. On the Add Constraint Mode window, type default in Name
- 3. Click on Add.
- 4. In the SDC Constraint File window, go to ./input_files and select router_LAXYZ_sdc
- 5. Click Open (SDC Constraint File window) and then OK (Add Constraint Mode window)

Amplueie Monor Liet	MMMIC Objects	Wizard Help
Analysis Views Analysis Views Analysis Views Hold Analysis Views Analysis Views Analysis Views Analysis Views Analysis Views Ormer: def Delay Corner: def OK Apply	alysis View fault close Help 3	This wizard will assist you in specifying the necessary information to configure the system for RC extraction, delay calculation, and timing analysis. It you have all the necessary data available, it is recommended that you configure the system as completely as possible for all steps of the implementation flow - through signoff. If not, you can always update the configuration, if necessary, as you proceed through the flow. If you are comfortable using the MIMMC Browser, you can use the Wizard Off button to remove the help dialog, and proceed at your own pace. For additional assistance with design import, press the Next button
		Prev

- 1. Double click on Analysis Views in the MMMC browser window
- 2. On the Add Analysis View window, type default in Name
- 3. Click OK



- Double click on **Setup Analysis Views** in the *MMMC browser* window
- In the Add Setup Analysis View window, make sure that Analysis View is set to 2. default 72
- Click **OK** 3.
Step 1: Import Design e- Analysis Configuration



- 1. Double click on **Hold Analysis Views** in the *MMMC browser* window
- In the Add Hold Analysis View Window, make sure that Analysis View is set to 2. default
- Click **OK** 3.



Step 1: Import Design e- Analysis Configuration

	er	
Analysis View List	MMMC Objects	Wizard Help
 ⊡- Analysis Views ⊕- default ⊕- Setup Analysis Views ⊕- default ⊕- Hold Analysis Views ⊕- default 	⊡ Library Sets ⊡ default ⊡ RC Corners ⊡ OP Conds ⊡ Delay Corners ⊡ default	This wizard will assist you in specifying the necessary information to configure the system for RC extraction, delay calculation, and timing analysis.
	😣 🗈 Save MMMC View Definition File	
	Look in: 2 Anome/zxp035/3D_0NoC/P&R/input	_files 🔽 🕒 🗇 🀑 🛅 📰 .
	Comput	4
	File name: Default.view	Save
1	Files of type: MMMC View Definition File (*.view*)	Cancel
<u>S</u> ave&Close	Load Delete <u>R</u> eset <u>P</u> references.	<u>W</u> izard Off <u>C</u> lose <u>H</u> elp

- 1. Click on Save&Close... in the MMMC browser window
- 2. Go to ./input_files
- 3. Type **Default.view** in File name
- 4. Click Save



Step 1: Import Design f- Result



In the welcome screen, we can see the modules of 3D-ONoC router before placement: 7 input_ports: (ip[0~6]), Switch_allocator (sw_allc), and Crossbar (cbar)



Step 1: Import Design g- Checkpoint

	😣 🖻 💷 Save Design		
	Data Type: Encounter OA		
	File Name: checkpoints/import.enc		
	Hierarchical DB	😣 🗈 Save Design	
5	<u>QK Apply Cancel H</u> elp	Look in: 2 Anter 2 Anter 2 Look in: 2 Anter 2 Comput] 📰
1.	Click in <i>File name</i>		
2.	Go to ./checkpoints		
3.	Type import.enc in <i>File name</i>		
4.	Click Save		
5.	Click OK		4
		File name: import.enc 3	<u>S</u> ave
		Files of type: All Files (*)	lancel

We should save the progress at each step. Click File-> Save Design



Step 2: Floorplan

- 1. Check Die Size by:
- 2. Enter **300** for both *Width* and *Height*
- 3. Enter 15 for
 - Core to Left
 - Core to Right
 - Core to Top
 - Core to Bottom
- 4. Click OK

sign Din	iensions				
ecify By:	🖲 Size 🔾 Die	e/IO/Core Co	ordinates		
🔾 Core	Size by: 💿 As	pect Ratio:	Ratio) (H/W);	525343439
			Core Uti	lization;	0.699994
			🔾 Cell Uti	lization;	0.699994
	🔾 Dir	mension;		Width:	105.55
<u> </u>				Height:	103.6
🖲 Die Si	ze by:		2	Width:	300
			2	Height:	300
Core Ma	rgins by: 🥑 Co	re to IO Bour	ıdary		
	O Co	re to Die Bou	ndary		
3	Core to Left	: 15	Core	e to Top:	15
•	Core to Right	: 15	Core to	Bottom:	15
Die Size	Calculation Use	: 🔾 Max I	0 Height	🥑 Min I	0 Height
			Toff Course		ntor

In this step we specify the floorplan Click Floorplan-> Specify Floorplan



Step 2: Floorplan

Vjew Partitio <u>n</u> Floorpl <u>a</u> n	Po <u>w</u> er <u>P</u> lace	Optimize <u>C</u> lock <u>R</u> oute]	Iming Verify Options Tools Flows Help
) 🥱 🕐 🕕 🍕	- Q G	(a, 🔣 a, O) ,	🌢 🕆 🔝 🗛 🗉 🚴 🛯 🖀 🔮
🦄 🔛 📓 🦋 🍓		🍓 🗓 1 🖳 🖳	V = 1 to the second sec
	TTTT	717-70 004	
	10	10-70000	
	il[0].ip	il[]].ip	
TU = 70.0%	TU=70.0%	TU=70.0%	
il[2].ip	h1[3].1p	h][4].ap	
10=20.085	J == 70.095	711-70.004	
1][5].p	ólap	sw alle cha	

1. In the main window, the boundaries of the chip appear.

2. Save your design under floorplan.enc in ./checkpoints directory



Step 3: Power Ring

- 1. Input **VDD VSS** in *Nets(s):*
- 2. Change the layer to:
 - metal 10 V for Top and Bottom
 - metal 9 H for *Left* and *Right*
- 3. Change the Width to 4
- 4. Change the Spacing to 2
- 5. Check Center in channel
- 6. Click OK

	VDD VSS					
Ring Ty	pe					
🖲 Core	ring(s) conta	ouring				
🖲 A	round core b	oundary	🔾 Along I/	O boundary		
E	wlude select	ed objects				
 Block 	ring(s) arou	ınd				
• E	ach block					
○ E	ach reef					
O Se	elected powe	er domain/fend	es/reefs			
⊖ E	ach selected i	block and/or g	group of core ro	17478		
0 C	lusters of sel	ected blocks ar	nd/or groups o	f core rows		
	With shar	ed ring edges				
🔾 User (defined coor	dinates:				MouseClick
۲	Core ring	 Block rit 	πg			
Ring C	onfiguration	1				
Interv	Top:	Bottom:	Left:	Right:	2	
Width	A		A HIELALY	A Hietaiyi		
to icatini	-				Undat	ha
Spacing		1) 5	2	2		
Spacing	Center		Specify	~	1	
Spacing: Offset:	0.005	0.095	0.095	0.095		
Spacing Offset:	0.095					
Offset: Offset	Set					
Spacing Offset: Option	Set			ate Basic		
Offset: Offset: Option	Set ption set:	•		ate Basic		



Step 3: Power Ring

r <u>P</u> lace <u>O</u> ptimize <u>C</u> lock	: <u>R</u> oute <u>T</u> iming	<u>V</u> erify Opt <u>i</u> ons Too	s Flow <u>s</u>	<u>H</u> elp			
Q Q Q 🕅 (R 🕑 👶 🕏		816	' 🖺 🕥 😭			
🐞 🕮 🗞 📆 1.	🖳 🛄 🖌	😐 🔩 🚯 🖏					
							7777288
	TU=70.096	TU=70.0%					
	il[[0]].tp	u[[1]].lp					
TU=70.0%6	TU=70.0%	TU=70.0%					
11[2]-1p TU=20.095	11[3] ap J==70.0\$6	nl(4) ap					
	1	TUI=70.0%6					
nl[5].ap. nl[i	6).aps	w_ale eba					
rt multiple objects							

In the main window, the power ring appears



Step 4: Power Stripe

Add Stripes

- 1. Input VDD VSS in Nets(s):
- 2. Change Layer to metal 8
- 3. Set:
 - Width to **4**
 - Spacing to 2
- 4. Change Set-to-set- distance to 50
- 5. Change the *Relative from core or* selected area: X from Left to **35**
- 6. Click OK

Ser Comi	juration
Net(s):	VDD VSS 1
Layer:	metal8 > Z
Direction:	• Vertical • Horizontal
Width:	4 3
Spacing:	2 Update
Set Patter	n
Set-to-s	et distance: 50 4
O Number	of sets:
O Bumps	🖲 Over 🕓 Between
Over P/	G pins 🛛 Pin layer: 🛛 Top pin layer 🕨 🗌 Max pin width: 🛛 0
Mas	ter name: O Selected blocks O All blocks
Strine Bo	mdary
Core rit	
O Pad rins	z 🔾 Inner 💿 Outer
🔾 Design b	νουπdary 🗹 Create pins
Each sel	ected block/domain/fence
🔾 All dom	ains
O Specify	rectangular area
O Specify	rectilinear area
First/Last	Stripe
Start from:	💿 left 🔾 right
• Relative	from core or selected area
X fron	1 left: 35 X from right: 0
🔾 Absolut	e locations
Option Se	t
Uro onti	ion set:
_ Ose opt	Un set.

Click on Power-> Power Planning->Add Stripe ...



Step 4: Power Stripe

r <u>P</u>lace <u>Optimize</u> <u>Clock</u> <u>Route</u> <u>Timing</u> <u>V</u>erify Options Tools Flows <u>H</u>elp



In the main window, the power stripes appears



Step 5: Power Routing

- Click on Route> Special Route -
- Input VDD VSS in Nets(s):
- Click **OK**

Block Pins 🗹 Pad Pins	🗹 Pad Rings 🗹 Follow Pins	Secondary Power Pins
Routing Control		
Top Layer: metal10	Bottom Layer: me	tall 🕨
Allow Jogging	🗹 Allow Layer	Change
X1: X2: Connect to Target In	Y1: Draw Y2: View Area side The Area Only	Ower Domain Selection All Selected Named:
Delete Existing Routes		
Generate Progress Messa Extra Config File:	ges	Extra Config Editing
		Target Editing Options



Step 5: Power Routing

					/ 198				6 AD / / / / / / / / / / / / / / / / / /	70	· · · · · · · · · · · · · · · · · · ·		 (/ / P
								<u></u>				1	調
				r ž	2 Z	Ø		Ø		掍		Ź	
						- A		Z		16	7	ź	
					48		8	Ø	Ź – Ż	10		Ź	
					23	E E	2	Ø			2	X	
				2	28		Ź	Z		46		Ŕ	86
					42	4	2	Ź.	2	扣	2 X	Z	8.0
					22	E E	Ź	Z	E E	14	2		
				×	4B		2	Ź		42	5 X	Ø	
					314		4	Z	Ka E	34	I I	Ø	
					42	E A	Ź	¥.		46		É	
					₹₿	Ŕ	Ź	Ź	A F	44		Ø	
					42	Ŕ	2	Ø	E E	412		Ø	
			121		48	¥		A		1		×.	
			1814 1814		₽₽	E E	Ź	XX		3	A A	Ŕ	
				7	42		2	Ø	g k	48	A	Z	
				ź	32	¥	Ź	Ø				¥	
					4 Z	2	2	Ø	E E	46	A A A A A A A A A A A A A A A A A A A	8	
					4£	3	4	Ø	ži – ž	16	1 8	Ø	
					22	Z	Ź	Ø	2		A N	Z	
					ąΖ	Ŕ	2	Ø	Z K	46		Z	
					2 Ź	Z	Á.	N	4	91		Ø	
			11		ŹŻ	Ź	Z	Ø				ž	
					9ģ	3	X	Ø	2	4 È	4 8	Ź	
				5	ŹŹ		2	Ź.	2			Ż	
						- A		Ø		16	7	ź	
	TU=70.096	TU=70.0%6			48		Ø	Ø	Ź – Ż	10		Ź	
						4	2	Ŷ.	2			X	
					28		Ž	Ź.	8	41		Ŕ	
					7 Ø		8	Ø	8		A K	X	
	il[0].ip	il[1].ip					Z	Z				¥.	
awa					48		E	Ź		4 E		NN	
10=>0.0~0	10=20.055	10=20.000			36	E E	4	Z		3 K	Z Z	Ø	
					组合		Z	Ź		12		Z	
					38	8		Ż	K K	24	Z Z	Ŕ	
					42	- A	2	Ø	Ŕ	48	8	Z	
II[2].ip	n][3].np	h][4].ip			39	E E	Ź	Ż				Ħ	
TTT70 006	TTT70.066			7	άŻ	t i i i i i i i i i i i i i i i i i i i	2	Ø		48		R	
1.5	1.0				38	2	2	Ø.	2	16	3	Ħ	
		TU=70.0%6		7				Ø				2	
					22	- Z	Ø	Ź			1 d	Ø	
					31/2		(<u>x</u>	1 8/	///////////////////////////////////////	ų	X/////////////////////////////////////	2	22
1][5].1p	1[6].ip	sw_alle lebai			//間			4		<u>/</u> #		1 ¹	

1. In the main window, the power routing appears

2. Save your design under power.enc in ./checkpoints directory



Step 6: Placement

😣 🖻 🗊 Place
💿 Run Full Placement 🔾 Run Incremental Placement 🔾 Run Placement In Floorplan Mode
Optimization Options
✓ Include Pre-Place Optimization □ Include In-Place Optimization
Number of Local CPU(s): 1 Set Multiple CPU
OK Apply Mode Defaults Cancel Help

Now we place the 3D_ONoC modules on the die:

- Click Place-> Place Standard Cell.
- Click OK



Step 6: Placement



In the main window, click on 🔝 to view the placed modules



Step 6: Placement



In the main window, click on 🏢 for the physical view



Step 7: Clock Tree a- Synthesize

😣 🖻 🗊 Synthesize Clock Tree	S 🖨 🗊 Generate Clock Spec
Basic Advanced 1	Specify Buffer/Inverter
Clock Specification Files: Clock.ctstch Gen Spec	Cells List Selected Cells
Results Directory: clock_report	BUF_X1 Add CLRBUF_X1 BUF_X16 Add CLRBUF_X1 BUF_X2 CLRBUF_X2 CLRBUF_X2 BUF_X32 BUF_X4 Delete BUF_X8 CLRBUF_X1 CLRBUF_X2 CLRBUF_X1 CLRBUF_X2 CLRBUF_X2 Output Specification File: Clock.ctstch
OK Apply Mode Load Spec Clear Spec Cancel Help	OK Apply Clear Spec Close Help
4	3

- 1. Click on Gen Spec
- 2. Select CLKBUF_X1 CLKBUF_X2 CLKBUF_X3
- 3. Click OK (Generate clock spec window)
- 4. Click **OK** (Synthesize Clock Tree)



Step 7: Clock Tree b- Display

- 1. Check Clock Route Only
- 2. Check Display Clock Tree
- 3. Click **OK**

😣 🗖 🗊 Display Clock Tree						
Clock Selection						
 All Clock(s) 						
Selected Clock						
Route Selection						
○ Pre-Route						
Clock Route Only						
○ Post-CTS						
O Post-Route						
Display Selection						
Display Clock Tree						
All Level						
 Bottom Level (non-gated clock tree only) 						
 Selected Level (non-gated clock tree only) 						
1						
O Display Clock Phase Delay						
O Display Min/Max Paths						
OK Apply Cancel Help						

Click on Clock ->Display -> Display Clock Tree



Step 7: Clock Tree



1. In the main window, the clock tree appears

2. Save your design under clock_syn.enc in ./checkpoints directory



Step 8: Nano Route a- Setting

😣 🗐 🗊 NanoRoute				
Routing Phase				
🗹 Global Route				
Detail Route Start Iteration 0 End Iteration default				
Post Route Optimization 🔲 Optimize Via 💭 Optimize Wire				
Concurrent Routing Features				
🗹 Fix Antenna	Insert Diodes Diode Cell Name			
✓ Timing Driven	Congestion Timing Effort 5 S.M.A.R.T.			
🔲 SI Driven				
Post Route SI	SI Victim File 🖻			
🔲 Litho Driven				
🔲 Post Route Litho Repair				
Routing Control				
Selected Nets Only	Bottom Layer default Top Layer default			
🔲 ECO Route				
Area Route Area	Select Area and Route			
Job Control				
🗹 Auto Stop				
Number of Local CPU(s): 1				
Number of CUP(s) per Remote Machine: 1				
Number of Remote Machine(s): 0				
Set Multiple CPU				
OK <u>A</u> pply A	ttribute Mode Save Load Cancel Help			

Click on Route-> NanoRoute-> Route Check Time Driven and then click OK



Step 8: Nano Route b- Report

```
zxp035@zxp035:~/3D_ONoC/P&R
                  24212
#Max overcon = 6 tracks.
#Total overcon = 0.26%.
#Worst layer Gcell overcon rate = 0.00%.
#Cpu time = 00:00:02
#Elapsed time = 00:00:02
#Increased memory = 4.00 (Mb)
#Total memory = 299.00 (Mb)
\#Peak memory = 330.00 (Mb)
#Start Detail Routing.
#start initial detail routing ...
     number of violations = 1
#
#cpu time = 00:00:08, elapsed time = 00:00:08, memory = 305.00 (Mb)
#start 1st optimization iteration ...
     number of violations = 0
#
#cpu time = 00:00:00, elapsed time = 00:00:00, memory = 305.00 (Mb)
#Complete Detail Routing.
#Total number of nets with non-default rule or having extra spacing = 73
#Total wire length = 61799 um.
#Total half perimeter of net bounding box = 51563 um.
#Total wire length on LAYER metal1 = 7055 um.
#Total wire length on LAYER metal2 = 25250 um.
#Total wire length on LAYER metal3 = 20096 um.
#Total wire length on LAYER metal4 = 7654 um.
#Total wire length on LAYER metal5 = 1387 um.
#Total wire length on LAYER metal6 = 264 um.
#Total wire length on LAYER metal7 = 87 um.
#Total wire length on LAYER metal8 = 0 um.
#Total wire length on LAYER metal9 = 5 um.
#Total wire length on LAYER metal10 = 0 um.
#Total number of vias = 29452
#Up-Via Summarv (total 29452):
```

Above is a sample of the report generated after the Nano Route step. It gives information about the wire length, metal used, etc..



Step 9: Optimization a- setting

- 1. Check Post-Route
- 2. Check Hold
- 3. Click OK

S-D Optin	nization	
Design Stage		
O Pre-CTS	Post-CTS	Post-Route
Optimization Typ	e	
🗹 Setup	1	Hold
 Incremental 		
🖲 Design Rules Vi	olations	
🗹 Max Cap		
🗹 Max Tran		
📃 Max Fanout		
🗌 🗆 Include SI 🔄	Options	J
	<u>nly M</u> ode D	efault <u>C</u> lose <u>H</u> elp



Step 9: Optimization b- report

mode WNS (ns): TNS (ns): Ing Paths: All Paths:	all 9.137 0.000 0 917	reg2reg 9.137 0.000 0 861	in2reg 9.263 0.000 0 910	reg2out 9.599 0.000 0 7	in2out N/A N/A N/A N/A	clkgate + N/A N/A N/A	 +
WNS (ns): TNS (ns): ing Paths: All Paths:	9.137 0.000 0 917	9.137 0.000 0 861	9.263 0.000 0 910	9.599 0.000 0 7	N/A N/A N/A	N/A N/A N/A	•
				+	+	N/A +	 +
node	all	reg2reg	in2reg	+ reg2out	in2out	+ clkgate	+
WNS (ns):	-0.007	0.057	-0.007	0.348	N/A	N/A	
TNS (ns):	-0.007	0.000	-0.007	0.000	N/A	N/A	
ing Paths:	1	0	1	0	N/A	N/A	
	217	001	, <u>, , , , , , , , , , , , , , , , , , </u>			1 11/2	1
 + Nr	nets(terr	Real ns) Wors	st Vio	Total Nr nets(te	L erms)		
ļ	0 (0)	0.	.000	0 (0)			
n put	0 (0) 0 (0)	0.	000	0 (0) 0 (0)			
	WNS (ns): TNS (ns): ing Paths: All Paths: 	WNS (ns): -0.007 TNS (ns): -0.007 ing Paths: 1 All Paths: 917 Nr nets(term 0 (0) n 0 (0) put 0 (0) .300%	WNS (ns): -0.007 0.057 TNS (ns): -0.007 0.000 ing Paths: 1 0 All Paths: 917 861 Real +	WNS (ns): -0.007 0.057 -0.007 TNS (ns): -0.007 0.000 -0.007 ing Paths: 1 0 1 All Paths: 917 861 910 Real Real Nr nets(terms) Worst Vio 0 (0) 0.000 n 0 (0) 0.000 put 0 (0) 0	WNS (ns): -0.007 0.057 -0.007 0.348 TNS (ns): -0.007 0.000 -0.007 0.000 ing Paths: 1 0 1 0 All Paths: 917 861 910 7 Real Total Nr nets(terms) Worst Vio Nr nets(terms) 0 (0) 0.000 0 (0) n 0 (0) 0.000 0 (0) put 0 (0) 0 0 (0) .300%	WNS (ns): -0.007 0.057 -0.007 0.348 N/A TNS (ns): -0.007 0.000 -0.007 0.000 N/A ing Paths: 1 0 1 0 N/A All Paths: 917 861 910 7 N/A Real Total Nr nets(terms) Worst Vio Nr nets(terms) 0 (0) 0.000 0 (0) n 0 (0) 0.000 0 (0) put 0 (0) 0 0 0 (0) .300%	WNS (ns): -0.007 0.057 -0.007 0.348 N/A N/A TNS (ns): -0.007 0.000 -0.007 0.000 N/A N/A ing Paths: 1 0 1 0 N/A N/A All Paths: 917 861 910 7 N/A N/A Real Total +

Above is a sample of the report generated after the Optimization step. 94 It gives information about the Setup and Hold violations, used metal layers thikness, etc..



Step 10: Adding Fillers





Step 10: Adding Fillers



1. Save your final design under **final.enc** in **./checkpoints** directory

Step 11: Design checking a- Layout Vs. Schematic (LVS)

******* Start: VERIFY CONNECTIVITY ******* Start Time: Wed Jun 4 20:14:23 2014

Design Name: router_LAXYZ Database Units: 2000 Design Boundary: (0.0000, 0.0000) (250.0000, 250.0000) Error Limit = 1000; Warning Limit = 50 Check all nets **** 20:14:24 **** Processed 5000 nets (Total 6074) Time Elapsed: 0:00:01.0

Begin Summary Found no problems or warnings. End Summary

End Time: Wed Jun 4 20:14:24 2014 ******* End: VERIFY CONNECTIVITY ******* Verification Complete : 0 Viols. 0 Wrngs. (CPU Time: 0:00:00.2 MEM: 0.004M)

Report displayed on the terminal

😣 🗆 🗉 Verify Connectivity
Net Type
● All
🔾 Regular Only
Special Only
Nets
● All
○ Selected
O Named:
Check
✓ Open ✓ UnConnected Pin ✓ Unrouted Net
Geometry Loop Geometry Connectivity Keen Previous Results
TSVDie Abstract File
Verify Connectivity Report: ts/router_LAXYZ.conn.rpt
Report Limits
Error: 1000
Warning: 50
Set Multiple CPU
<u>OK</u> <u>Apply</u> <u>Cancel</u> <u>H</u> elp

Click Verify->Verify Connectivity

Save the Verify Connectivity Report.rpt under ./reports, and then click OK

Step 11: Design checking b- Design Rule Check (DRC)

🛞 🗆 🛛 Verify Geometry	
Secience Verification Area Entire area Specify X1: Y1: Y1: Y1: Y2: Y2: Check Minimum Width Minimum Area Short Cell Overlap Insufficient Metal Overlap Minimum Cut Minimum Cut Va Enclosure Number of Local CPU(s): 1	VERIFY GEOMETRY Cells : 0 Viols. VERIFY GEOMETRY SameNet : 0 Viols. VERIFY GEOMETRY Wiring : 0 Viols. VERIFY GEOMETRY Antenna : 0 Viols. VERIFY GEOMETRY Sub-Area : 1 complete 0 Viols. 0 Wrngs. VG: elapsed time: 3.00 Begin Summary Cells : 0 SameNet : 0 Wiring : 0 Antenna : 0 Short : 0 Overlap : 0 End Summary
✓ Via Enclosure Allow ✓ Pin In Blockage ✓ Same Cell Volations Different Cell Volations Overlap of Pad Filler Ce Overlap of Routing Bloc Owerlap of Routing Bloc Overlap of Routing Bloc Overlap of Routing Bloc Owerlap of Routi	Over tap . 0 End Summary Verification Complete : 0 Viols. 0 Wrngs. ************************************

Click Verify->Verify geometry

Under Advanced, save the Verify Geometry Report.rpt under ./reports. 98 Click OK



Step 12: Output files a- SPEF file

S - D Extract RC				
Save RC				
Save Cap to router_LAXYZ.cap	ð			
Save Setload to router_LAXYZ.setload	Þ			
Save Set Resistance to router_LAXYZ.setres	Þ			
Save SPF to router_LAXYZ.spf	Þ			
Save SPEF to router_LAXYZ.spef	Þ			
RC Corner to Output				
<u>OK</u> <u>Apply</u> <u>Cancel</u> <u>H</u> elp				

- 1. Click on Timing-> Extract RC
- 2. Check Save SPEF to
- 3. Click OK



Step 12: Output files b- SDF file

Son Calc	ulate Delay
Delay Calculat	ion Option
🗹 Ideal Clock	
SDF Output File:	reports/router_LAXYZ.sdf [🛅
	pply <u>C</u> ancel <u>H</u> elp

- 1. Click on Timing-> Extract RC
- 2. Save the SDF Output File in ./reports
- 3. Click OK



Step 12: Output files c- Netlist file

😣 🗆 🗉 Save Netlist	
☑ Include Intermediate Cell Definition	
Include Leaf Cell Definition	
Netlist File: reports/router_LAXYZ_final.vnet	7
OK Cancel Help	

- 1. Click on File-> Save-> Netlist
- 2. Save the router_LAXYZ_final.vnet in ./reports
- 3. Click OK



- The 12 steps previously presented can be made via commands inserted in the SoC Encounter terminal.
- The commands required for the Place&Route are grouped in a single .*tcl* file.
- The .*tcl* file is named "*par_LAXYZ.tcl*"
- It is located in:
 "/home/zxp035/3D-ONoC/P&R/script"
- To run the TCL script type on your terminal:

velocity 1> source /3D-NoC/P&R/script/par_LAXYZ.tcl



Script: par_LAFT.tcl (1/8)

```
#
# Step 1: Setup (File --> Import Design)
#
setUIVar rda_Input ui_netlist ./input_files/router_LAXYZ.vnet
setUIVar rda Input ui timingcon file ./ input files/router LAXYZ.sdc
setUIVar rda Input ui topcell router LAXYZ
setUIVar rda Input ui leffile ~/lib/NangateOpenCellLibrary.lef
setUIVar rda_Input ui_timelib ~/lib/typical.lib
setUIVar rda Input ui pwrnet VDD
setUIVar rda Input ui gndnet VSS
setUIVar rda Input ui_cts_cell_list {CLKBUF_X1 CLKBUF_X2 CLKBUF_X3}
commitConfig
```

Checkpoint

saveDesign import.enc



Script: par_LAFT.tcl (2/8)

```
#
# Step 2: Floorplan (Floorplan --> Specify Floorplan)
#
floorPlan -s 300 300 15 15 15 15
# Checkpoint
saveDesign floor.enc
#
# Step 3: Power ring (Power --> Power Planning --> Add Ring)
#
createPGPin VDD -net VDD
createPGPin VSS -net VSS
globalNetConnect VDD -type pgpin -pin VDD -sinst dp/rf
globalNetConnect VSS -type pgpin -pin VSS -sinst dp/rf
```



Script: par_LAFT.tcl (3/8)

```
addRing -nets {VSS VDD} -type core rings \
 -spacing top 2 -spacing bottom 2 -spacing right 2 -spacing left 2 \setminus
 -width top 4 -width bottom 4 -width right 4 -width left 4 \setminus
 -around core -jog distance 0.095 -threshold 0.095 \
 -layer top metal10 -layer bottom metal10 -layer right metal9 \
-layer left metal9 \
-stacked via top layer metal10 -stacked via bottom layer metal1
#
# Step 4: Power stripe (Power --> Power Planning --> Add Stripe)
#
addStripe -nets {VSS VDD} -layer metal8 -width 2 -spacing 1.5 \
 -block ring top layer limit metal9-block ring bottom layer limit metal7 \
 -padcore ring top layer limit metal9-padcore ring bottom layer limit metal7
 -stacked via top layer metal10 -stacked via bottom layer metal1
 -set to set distance 50 -xleft offset 50 -merge stripes value 0.095 \
 -max same layer jog length 1.6
```



Script: par_LAFT.tcl (4/8)

```
# Step 5: Power route (Route --> Special Router)
```

```
#
```

#

```
sroute -nets {VSS VDD} -layerChangeRange {1 10} \
```

```
-connect { blockPin padPin padRing corePin floatingStripe } \
```

```
-blockPinTarget { nearestRingStripe nearestTarget } \
```

```
-padPinPortConnect { allPort oneGeom } \
```

```
-checkAlignedSecondaryPin 1 -blockPin useLef -allowJogging 1 \
```

```
-crossoverViaBottomLayer 1 -allowLayerChange 1 -targetViaTopLayer 10 \
```

```
-crossoverViaTopLayer 10 -targetViaBottomLayer 1
```

```
# Checkpoint
saveDesign power.enc
#
# Step 6: Placement (Place --> Standard Cell)
#
placeDesign -prePlaceOpt
```



#

Script: par_LAFT.tcl (5/8)

```
# Step 7: Clock tree synthesis (CTS)
#a- Synthesis: (Clock --> Synthesize Clock Tree)
```

```
addCTSCellList {CLKBUF_X1 CLKBUF_X2 CLKBUF_X3}
clockDesign -genSpecOnly Clock.ctstch
clockDesign -specFile Clock.ctstch -outDir clock_report -fixedInstBeforeCTS
```

```
# Checkpoint
saveDesign clock syn.enc
```

```
#
```

```
# b- Display: (Clock --> Display --> Display Clock Tree)
This step should be done manually
```

#



Script: par_LAFT.tcl (6/8)

```
#
# Step 8: Detailed route (Route --> Nano Route --> Route)
#
setNanoRouteMode -quiet -routeWithTimingDriven true
setNanoRouteMode -quiet -routeTopRoutingLaver default
setNanoRouteMode -quiet -routeBottomRoutingLayer default
setNanoRouteMode -quiet -drouteEndIteration default
setNanoRouteMode -quiet -routeWithTimingDriven true
routeDesign -globalDetail
#
# Step 9: Optimization (postRoute) (Optimize --> Optimize Design)
#
optDesign -postRoute
```

optDesign -postRoute -hold

Checkpoint

saveDesign route.enc


Script: par_LAFT.tcl (7/8)

```
#
# Step 10: Add fillers (Place --> Physical Cells --> Add Filler)
#
addFiller -prefix FILLER -cell FILLCELL X1 FILLCELL X2 FILLCELL X4 \
 FILLCELL X8 FILLCELL X16 FILLCELL X32
#
# Step 11: Verification (LVS) (Verify --> Verify Connectivity)
#
verifyConnectivity -type all -error 1000 -warning 50 # LVS check
verifyGeometry # DRC
#
# Step 12: Data out (Timing --> Extract RC, Timing --> Write SDF, File --> Save --> Netlist)
#
saveNetlist router LAXYZ final.vnet # Netlist
```



Script: par_LAFT.tcl (8/8)

isExtractRCModeSignoff

rcOut -spef router_LAXYZ.spef # SPEF file

delayCal -sdf router_LAXYZ.sdf –idealclock # SDF file

Final checkpoint

save Design final.enc



<== Back to Contents

3. Design checking LVS (Layout-Versus-Schematic) & Design Rule Check (DRC)



Objectives

- In this tutorial, we check the correctness of the designed 3D-ONoC router. Two main checking process are performed in this tutorial:
 - Layout Versus Schematic (LVS)
 - Checks whether the integrated circuit layout in the Place & Route phase (Phase 2) corresponds to the original schematic or circuit diagram of the design obtained in the Design Synthesis phase (Phase 1).
 - Design Rule Check (DRC)
 - Ensures that the layout conforms to the rules designed/required for faultless fabrication.



Requirements

- Before starting the design check, you should have already finished the two previous steps:
 - Design synthesis (DS)
 - Place & Route (P&R)
- If you are not continuing the previous two steps, you need the final.enc file and file.enc.dat folder to be copied first to the ./checkpoints directory to restore the final post P&R design



Design Check directory structure





Contents

- Environment
- Restore design
- Layout Vs. Schematic (LVS)
- Design Rule Check (DRC)
- Commands



<pre>Ele Edit View Terminal Tabs Help [zxp035@zxp035 ~]\$ tcsh /home/zxp035% cd 3D_ONoC/ /home/zxp035/3D_ONoC% mkdir Design_check /home/zxp035/3D_ONoC/Design_check% mkdir checkpoint /home/zxp035/3D_ONoC/Design_check%</pre>											z	хрO)35	@z	кр (35:	~					ll×
[zxp035@zxp035 ~]\$ tcsh /home/zxp035% cd 3D_ONoC/ /home/zxp035/3D_ONoC% cd Design_check/ /home/zxp035/3D_ONoC/Design_check% mkdir checkpoint /home/zxp035/3D_ONoC/Design_check% ■	<u>F</u> ile	Ē	dit	⊻i	ew	Ţ€	erm	nina	al T	ā <u>b</u> s	He	elp										
	[zxp /hom /hom /hom /hom	003 ne/ ne/ ne/ ne/	5@z zxp zxp zxp zxp zxp	2×p 003 003 003 003	935 5% (5/31 5/31 5/31 5/31	~] D_C D_C D_C D_C	3D 3D 0No 0No 0No	tc:)_01)C%)C%)C/[)C/[sh NoC/ mkc cd Desi Desi	/ Des ign_ ign_	Des ign che che	ign _ch ck% ck%	_ch eck mk	neck K/	- cl	heck	poi	nt				

- Make sure that you are working under **cshr** environment. Otherwise type **tcsh**.

- Go to /home/zxp035/3D-ONoC/ and make Design_check
- In the new Design_check directory, make a new directory checkpoint



zxp035@zxp035:~	
<u>F</u> ile <u>E</u> dit <u>V</u> iew <u>T</u> erminal Ta <u>b</u> s <u>H</u> elp	
<pre>[zxp035@zxp035 ~]\$ tcsh /home/zxp035% cd 3D_0NoC/ /home/zxp035/3D_0NoC% cd Design_check/ /home/zxp035/3D_0NoC/Design_check% mkdir checkpoint /home/zxp035/3D_0NoC/Design_check% cp/PandR/checkpoints/final.enc ./checkpoint/ /home/zxp035/3D_0NoC/Design_check% cp -r/PandR/checkpoints/final.enc.dat ./checkpoint/ /home/zxp035/3D_0NoC/Design_check%</pre>	

In the new **checkpoint** directory, we will copy the last checkpoint performed₁₁₇ in P&R phase. We need to copy **final.enc** file and **final.enc.dat** folder



		zxp035@zxp035:~	
<u>F</u> ile	<u>E</u> dit <u>V</u> iew <u>T</u> erminal Ta <u>b</u> s <u>H</u>	lelp	
[zxp /hom /hom /hom /hom /hom	2cm _new _new _new main mg/s _ 35@zxp035 ~]\$ tcsh 2/zxp035/3D_0NoC% mkdir De 2/zxp035/3D_0NoC% cd Desig 2/zxp035/3D_0NoC/Design_ch 2/zxp035/3D_0NoC/Design_ch 2/zxp035/3D_0NoC/Design_ch	<pre>sign_check n_check/ eck% mkdir checkpoint eck% cp/PandR/checkpoints/final.enc.dat ./checkpoint/ eck% cp -r/PandR/checkpoints/final.enc.dat ./checkpoint/ eck% </pre>	
			v

Use cp command to copy final.enc file from ../PandR/checkpoints into ./checkpoint Use cp – r command to copy also final.enc.dat folder



zxp035@zxp035:~	
<u>F</u> ile <u>E</u> dit <u>V</u> iew <u>T</u> erminal Ta <u>b</u> s <u>H</u> elp	
<pre>[zxp035@zxp035 ~]\$ tcsh /home/zxp035% cd 3D_ONoC/ /home/zxp035/3D_ONoC% mkdir Design_check /home/zxp035/3D_ONoC% cd Design_check/ /home/zxp035/3D_ONoC/Design_check% mkdir checkpoint /home/zxp035/3D_ONoC/Design_check% cp/PandR/checkpoints/final.enc ./checkpoint/ /home/zxp035/3D_ONoC/Design_check% cp -r/PandR/checkpoints/final.enc.dat ./checkpoint/ /home/zxp035/3D_ONoC/Design_check% velocity</pre>	



Restore design

🗙 🗖 🔲 Restore Design		
Data Type: 🔍 Encounter 🕓 OA	1	
Restore Design File: checkpoint/final	ial.enc 🖻	
Sync Relative Path		
With ERROR Line messages		
🔲 With WARN messages 🛛 🔍	😣 🗉 Restore Design	
	Look in: 2 📄 /home/zxp035/3D_0NoC/Design_check/checkpoint 🔽 📀 🐑 😁 🖄	::
	Comput	
1. Click on the folder		
2. Go to ./checkpoint		
2 Soloct final one file		
4. Click Open		
5. Click OK		
	4	•
F	File name: final.enc	en
F	Files of type: Design files (*.enc)	icel

First, we should restore the final design of the P&R phase Click File>Restore design



Restore design



The final layout of the P&R phase should appear

Layout Vs. Schematic (LVS)

******* Start: VERIFY CONNECTIVITY ******* Start Time: Wed Jun 4 20:14:23 2014

Design Name: router_LAXYZ Database Units: 2000 Design Boundary: (0.0000, 0.0000) (250.0000, 250.0000) Error Limit = 1000; Warning Limit = 50 Check all nets **** 20:14:24 **** Processed 5000 nets (Total 6074) Time Elapsed: 0:00:01.0

Begin Summary Found no problems or warnings. End Summary

End Time: Wed Jun 4 20:14:24 2014 ******* End: VERIFY CONNECTIVITY ******* Verification Complete : 0 Viols. 0 Wrngs. (CPU Time: 0:00:00.2 MEM: 0.004M)

Report displayed on the terminal

	8 – D Verify Connectivity
	Net Type
	● All
	🔾 Regular Only
	Special Only
	Nets
)	● All
	○ Selected
	O Named:
	Check
	🗹 Open 🗹 UnConnected Pin 🗹 Unrouted Net
	🔲 Connectivity Loop 🗹 DanglingWire (Antenna) 🗹 Weakly Connected Pin
	Geometry Loop Geometry Connectivity Keep Previous Results
	TSV Die Abstract File
	Verify Connectivity Report: ts/router_LAXYZ.conn.rpt
	Report Limits
	Error: 1000
	Warning: 50
	Set Multiple CPU
	OK Apply Cancel Help

Click Verify->Verify Connectivity

Save the Verify Connectivity Report.rpt under ./reports, and then click OK



Design Rule Check (DRC)

Second Verify Geometry		
Solution Verify Geometry Basic Advanced Verification Area Entire area Entire area Specify Specify Prave X1: O Y1: X2: O Y2: Check Minimum Width Minimum Area Image: Short Cell Overlap Image: Short Cell Overlap Image: Short Minimum Cut Image: Short Overlap of Pad Piller Cell Overlap of Routing Bloc Overlap of Routing Bloc Overlap of Routing Bloc Overlap of Routing Bl	Verify Geometry Basic Advanced Verification Area Regular Routing Only Stacked Vas On Regular Routing Only Vire Extension Via Overlap Vas Default Rule Maximum Width Max Nonpreferred Wire Length: 0 Number of Local CPU(s): 1 Set Multiple CPU Number of Local CPU(s): 1 Set Multiple CPU Number of Local CPU(s):	<pre>VERIFY GEOMETRY Cells : 0 Viols. VERIFY GEOMETRY SameNet : 0 Viols. VERIFY GEOMETRY Wiring : 0 Viols. VERIFY GEOMETRY Antenna : 0 Viols. VERIFY GEOMETRY Sub-Area : 1 complete 0 Viols. 0 Wrngs. VG: elapsed time: 3.00 Begin Summary Cells : 0 SameNet : 0 Wiring : 0 Antenna : 0 Short : 0 Overlap : 0 End Summary Verification Complete : 0 Viols. 0 Wrngs. ******End: VERIFY GEOMETRY******* *** verify geometry (CPU: 0:00:02.1 MEM: 83.5M)</pre>
	<u>QK</u> <u>Apply R</u> eset <u>C</u> ancel <u>H</u> elp	

Click Verify->Verify geometry

Under Advanced, save the Verify Geometry Report.rpt under ./reports. Click OK



Commands

• To perform the Layout Vs. Schematic (LVS) type the following command on your terminal:

velocity 1> verifyConnectivity -type all -error 1000 -warning 50

To perform the Design Rule Check (DRC) type the following command on your terminal:

velocity 2> verifyGeometry



<== Back to Contents

4. Post-Layout simulation



Objectives

- After completing this tutorial you will be able to:
 - Check if the post-layout design is free from any timing violations
 - Report timing and area
 - Evaluate the power consumption (dynamic and static)
 - Learn how to make the post-layout simulation via:
 - The CAD Graphic User Interface
 - Tcl script



Contents

- Requirements
- Post-layout simulation directory structure
- Setup
- Post layout synthesis (Step 1~3)
- Script



Requirements

- Before starting the post-layout, you should have already finished the three previous phases:
 - Design Synthesis (DS)
 - Place & Route (P&R)
 - Design Check (LVS and DCR)
- We should create a new directory: ~/3D-ONoC/Post

where the post-layout simulation is performed.



Post-layout directory structure





- Before we start, we should copy some output files that we will use for this phase.
 - router_LAXYZ_final.vnet (From P&R phase)
 - router_LAXYZ_final.spef (From P&R phase)
 - router_LAXYZ_final.sdf (From P&R phase)
 - router_LAXYZ_final.sdc (From Synthesis phase)
- These files should be copied to ~/3D-ONoC/Post/input



Setup

Zxp035@zxp035:~	
<u>F</u> ile <u>E</u> dit <u>V</u> iew <u>T</u> erminal Ta <u>b</u> s <u>H</u> elp	_
<pre>[zxp035@zxp035 ~]\$ tcsh /home/zxp035% cd 3D_ONoC/ /home/zxp035/3D_ONoC/Post% cp/Synthesis/output_files/router_LAXYZ.sdc ./input/ /home/zxp035/3D_ONoC/Post% cp/PandR/router_LAXYZ.sdf ./input/ /home/zxp035/3D_ONoC/Post% cp/PandR/router_LAXYZ.sdf ./input/ /home/zxp035/3D_ONoC/Post% cp/PandR/router_LAXYZ_final.vnet ./input/ /home/zxp035/3D_ONoC/Post% []</pre>	
	Ţ
Type tcsh and go to ~ 3D-ONoC/Post	
Using the "cp" command, copy the necessary four files to .	/input
Start Design Compiler by typing design_vision	



Step 1: Timing analysis

- For this step, we execute a script that contains the necessary operations for time analysis.
- The operations are almost the same as the ones performed in Phase 2 (Design Synthesis) of this tutorial.
- The *sta_LAXYZ.tcl* script file needed for this step is located in ./scripts
- Next slides depicts sta_LAXYZ.tcl



Step 1: Timing analysis: sta_LAXYZ.tcl (1/3)

Define the variable which we will use

<pre>set base_name</pre>	"router_LAXYZ"
<pre>set vnet_file</pre>	"router_LAXYZ_final.vnet"
<pre>set spef_file</pre>	"router_LAXYZ.spef"
<pre>set sdf_file</pre>	"router_LAXYZ.sdf"
<pre>set sdc_file</pre>	"router_LAXYZ.sdc"

Step 1: Set the libraries:

set target_library "~/lib/typical.db"
set synthetic_library "~/lib/dw_foundation.sldb"
set link_library [concat "*" \$target_library \$synthetic_library]
set symbol_library ""~/lib/generic.sdb"
define_design_lib WORK -path ./WORK # redirect the log files to a new folder "WORK"



Script: sta_LAXYZ.tcl (1/2)

Step 2: Read post_layout netlist####
read_file -format verilog ./input/\$vnet_file
current_design \$base_name
link

Delay and RC information####
read_sdc ./input/\$sdc_file
read_sdf ./input/\$sdf_file
read_parasitics ./input/\$spef_file

Generate reports####

report_timing > ./reports/timing_report_\${base_name}.txt
report_reference -hier > ./reports/reference_report_\${base_name}.txt



Step 1: Timing analysis Execute the script

😣 🗊 Execute Script File
Look in: 🔄 /home/zxp035/3D_ONoC/Post/scripts/
sta_LAXYZ.tcl
File <u>n</u> ame: sta_LAXYZ.tcl Open
File type: Script Files (*.script *.scr *.dcs *.dcv *.dc *.dcfpga *.con *.tcl *.tcl) Cancel
Echo commands
✓ Verbose

To run the TCL script, click File> Execute script Go to ./scripts, select sta_LAXYZ.tcl and click Open

Step 1: Timing analysis Reports

* Some/all delay information is back-annotated.

perating conditions, cypical hibrary, wangateopencerinibiary					
/ire Load Model Mode: top		1.330000	2	2.660000	
	OR3_X1 NangateC	DpenCellLibrary			
Startpoint: cbar/cntri_reg_reg[41]		1.330000	1	1.330000	
(rising edge-triggered flip-flop clocked by clk)	1				
Endpoint: 11[3].1p/sw_req_reg	Total 8 references			19.684000	
(rising edge-triggered flip-flop clocked by clk)					
Path Group: CIK	****	*****			
Path Type: max	Design: stop go 6				
Deg/Clust/Dert Nire Lead Medel Library	****	*****			
Des/Clust/Port Wire Load Model Library	Deference Library	Unit Area	Count	Wetal Area	Attack
router LANVZ 5K buratio 1.1 NangatoOpenCellLibuary	Reference Library	UNIC Area	Counc	TOLAI Area	ACCLIDUC
router_making Sh_Hvidtio_1_1 Mangateopencerinibidiy	Notall Va				
Point Incr. Path	A01211_X2 Nangated	pencellLibrary			
		2.394000	1	2.394000	
clock clk (rise edge) 0.00 0.00	DFF_X1 NangateC)penCellLibrary			
clock network delay (ideal) 0.00 0.00		4.522000	2	9.044000	n
cbar/cntrl reg reg [41]/CK (DFF X1) 0.00 0.00 r	INV_X1 Nangated	DpenCellLibrary			
cbar/cntrl reg reg[41]/0 (DFF X1) 0.09 * 0.09 f		0.532000	2	1.064000	
cbar/FE OFC71 cntrl reg 41 /Z (CLKBUF X3) 0.11 * 0.20 f	NAND2_X1 NangateC)penCellLibrary			
cbar/output loop[5].cbar mux/cntrl[6] (mux out n in7 WIDTH34 1)		0.798000	1	0.798000	
0.00 0.20 f	NAND3 X1 Nangated	DpenCellLibrary			
cbar/output_loop[5].cbar_mux/U31/ZN (NOR2_X1) 0.07 * 0.27 r		1.064000	2	2,128000	
cbar/output_loop[5].cbar_mux/U21/ZN (INV_X1) 0.02 * 0.29 f	NAND4 X1 NapgateC	penCellLibrary			
cbar/output_loop[5].cbar_mux/U25/ZN (NOR3_X1) 0.08 * 0.36 r	In interior in interior	1 330000	1	1 330000	
cbar/output_loop[5].cbar_mux/U44/ZN (AND4_X2) 0.15 * 0.51 r	Naprato	nonCollLibrary	1	1.550000	
cbar/output_loop[5].cbar_mux/U24/ZN (AOI222_X1) 0.06 * 0.57 f	Nangated	pencerimprary	2	2 660000	
cbar/output_loop[5].cbar_mux/U150/ZN (NAND3_X1) 0.09 * 0.67 r		1.330000	2	2.660000	
<pre>cbar/output_loop[5].cbar_mux/data_out[0] (mux_out_n_in7_WIDTH34_1)</pre>	Nangated	pencellLibrary			
0.00 0.67 r		1.330000	1	1.330000	
cbar/data_out[170] (crossbar_NOUT7_NIN7_WIDTH34) 0.00 0.67 r					
sw_allc/tail_sent[5] (sw_alloc_NOUT7) 0.00 0.67 r	Total 8 references			20.748000	
sw_allc/U215/ZN (OAI211_X1) 0.05 * 0.71 f	1				
sw_allc/U214/ZN (INV_X1) 0.06 * 0.78 r	design_vision>				
<pre>sw_allc/ol[5].mat_arb/request[2] (matrix_arb_formultistage_SIZE7_1)</pre>					
0.00 0.78 r	Log History				
Ig History	design vision>				

Timing report

Reference report

Analysis reports for timing and area will be saved in *./reports* 136 The reports contain detailed evaluation of reference (area) and timing delay by module



Step 2: Timing simulation

- In this second step, we will check whether our design is free from any delay violation.
- We will use *ncverilog* and *simvision*.
- We created a test bench file, named *Test.v*, in order to evaluate 3D-ONoC router.
- In this test bench, random flits are generated, injected from the 7 input-ports of the router and ejected from the 7 output ports.
- Finally, the correctness of the ejected flits is checked.

Step 2: Timing simulation: a- Hierarchy



Test.v is located in *./verilog_src* along with the remaining Verilog source files

138

Step 2: Timing simulation: b-*Test.v* (1/4)

`timescale 1	lns/1ns
module Test	t;
reg	clk;
reg	reset;
//wire list of	bject value output variables
wire [37:0] wire [6:0]	dat_out_local, dat_out_north, dat_out_east, dat_out_south, dat_out_west, dat_out_up, dat_out_down; stop_out;
//register lis	st object value of input test
reg [37:0]	dat_in_local, dat_in_north, dat_in_east, dat_in_south, dat_in_west, dat_in_up, dat_in_down;
//*******	
//*******	registers used for the payload of input data ******//
reg [20:0]	payload_1;
reg [20:0]	payload_2;
reg [20:0]	
reg [20:0]	payload_4;
reg [20:0]	payload_5;
reg [20:0]	payload_6;
reg [20:0]	payload_7;

Step 2: Timing simulation: b-*Test.v* (2/4)

//Top module definition

router_LAXYZ router (.clk(clk),

.reset(reset),

.data_in({dat_in_down, dat_in_up, dat_in_west, dat_in_south, dat_in_east, dat_in_north, dat_in_local}), .data_out({dat_out_down, dat_out_up, dat_out_west, dat_out_south, dat_out_east, dat_out_north, dat_out_local}),

.stop_in({stp_in_local, stp_in_norh, stp_in_east, stp_in_south, stp_in_west, stp_in_up, stp_in_down}), .stop_in(7'b0000000),

.stop_out(stop_out), .xaddr(3'b010),.yaddr(3'b010),.zaddr(3'b010)); // We assume that the router has 222 adress

```
//clock generation (100 Mhz frequency )
```

```
always #5000 clk = ~clk;
```

```
//Annotation file initialization
```

initial begin

```
`ifdef ___POST_PR___
```

\$sdf_annotate("input/router_LAXYZ.sdf", Test.router, , "sdf.log", "MAXIMUM");

`endif

#0

clk = 1;

reset = 1'b1;

Step 2: Timing simulation: b-*Test.v* (3/4)

```
#100000
//Initialization of the vcd file that collects simulation information
   $dumpfile("dump.vcd");
   $dumpvars(0, Test);
//Start sending flits
              for(i=0;i<100;i=i+1)begin //We assume the number of sent flit is 100 for simplicity
              #10000
              //*** local port sending
              if (stop out[0] == 1)
                             dat in local = 0;
              else begin
                             dat in local = {payload 1,9'b010011010,7'b0000010,1'b1};//(0,1)
                             payload 1 = payload 1 + 1;
                             sent1 = sent1 + 1;
               end
          //*** We perform the same operations for the remaining input-ports
```

Step 2: Timing simulation: b-*Test.v* (4/4)

end//for loop end

#100000

\$finish;

end // initial begin

always @(dat_out_local) begin // Count the flit received at the local out-port

rec1= rec1+1;

end

rec5+1;

end

always @(dat_out_north) begin// Count the flit received at the north out-port rec2= rec2+1;

end

•••••// Count the flit received at the remaining out-ports

always @(dat_out_down) begin

rec7= rec7+1;

end

endmodule // The end of Test.v

Step 2: Timing simulation: c- Compilation

- Using ncverilog we compile our test bench Test.v with our top module netlist file router_LAXYZ_final.vnet
- The result of this compilation is the **dump.vcd** file previously initialized in *Test.v*
- In your terminal and under ~/3D-ONoC/Post type the following command:

/home/zxp035/3D_ONoC/Post% ncverilog +access+r +define+__POST_PR__
verilog_src/Test.v input/router_LAXYZ_final.vnet -v ~/lib/typical.lib

Step 2: Timing simulation: d- Simulation

- Now, we launch **simvision** to see the result of the simulation
- In your terminal and under ~/3D-ONoC/Post type the following command:

/home/zxp035/3D_ONoC/Post% simvison &
🙁 🗖 🔲 Design Browser 1 - SimVision
Eile Edit Yiew Select Explore Windows Help
📔 🧬 중 🛛 🗠 🔍 🛝 🛍 🗙 🥜 🔤 🍣 - 🗊 🧼 Send To: 💽 🧱 🖹 🗱 🐯 🌄 📟
Design Browser × ① Name ▼
Browse: O All Available Detions
Leaf Filter: *
Show contents: In the signal list
Show concents. In the signal list
0 objects selected

Simvision welcome screen Click on File> Open Database ...

🛛 🖨 🔲 Design Browser 1 - Sir	mVision	
Eile Edit Yiew Select Explo	ore Windows	Help
🗐 🖨 충 📔 🗠 🖂 🗼 🛍	🖹 🗙 🧷 🥵 - 🗊 🔶 🦉 Send To: 💽 🧱 🖹 👔	: 🔯 🌄 📰
Design Browser	× ③ Name ×	
Browse: 🧿 All Available 🗹 🏢	Options	
_		
	🗖 🗉 Open Database	
Dire	ectory: /home/zxp035/3D_ONoC/Post 🗕 主	1
	.simvision INCA_libs input reports scripts verilog_src dump.vcd File name: dump.vcd	
File	es of type: VCD Files (*.vcd)	
Leaf Filter: *	Transition Files (*.trn)	
	Transition & Design Files (*.trn,*.dsn)	
Show contents: In the s	signal list VCD Files (*.vcd)	*
	All Files (*)	<u> </u>
()	, do 0	jects selected

Change Files of type to VCD files (*.vcd) Select dump.vcd Click Open

🙁 🗕 💷 Design Browser 1 - SimVision
Eile Edit Yiew Select Explore Windows Hel
🔚 🔐 🔗 🗠 🖙 👔 🍋 🛍 🗙 🥜 🔤 🏶 🐨 🗛 🖉 👫 ன 🐨
ox Web Browser × ④ Name ▼
Browse: 🔵 All Available 🚽 📖 Options
😣 🔿 💿 File Translation
The file you have selected is a VCD file.
It will automatically be translated into an SST database.
Destination: /home/zxp035/3D_ONoC/Post/dump.t Browse
Duration: 🔹 All Time
✓ Range
Start Time: End Time:
Include Sequence Time Information
Compress Resulting Database
Leaf Filter: * OK Cancel Help
0 objects selecte

File Translation window will appear Click on **OK** to translate **dump.vcd** into **dump.trn**

dump.trn file will be used to visualize the test bench signals

Design Browser 1 - SimVision Eile Edit Yiew Select Explore Windows Help Send To: 🖹 🔚 🤯 📰 🗐 i 🚰 😽 St - 🥄 🚽 🖻 🐘 🗶 🥖 KO CX Х. Design Browser Name 🔻 💶 clk 🔂 rec2 Browse: 🦲 All Available 🕶 🔝 Options... 0 🚮 dat_in_down[37:0 🐻 rec3 dat_in_east[37:0] 📠 rec4 emub 📻 🗉 **77**) dat_in_local[3/:0] 🔂 rec5 🗄 🛄 Test dat_in_north[67:0] 🔂 rec6 3 dat_in_south[37:0] 🔂 rec7 dat_in_up[**77:**0] 💶 reset dat_in_west[37:0] 🚮 sent 1 dat_out_down[37:0] ቬ sent2 dat_out_east[37:0] 🔂 sent3 Ŀ, adat_out_loca1[37:0] 🚮 sent4 adat_out_north[37:0] 💼 sent5 🔂 dat_out_south[37:0] ҧ sent6 🚮 dat_out_up[37:0] 🔂 sent7 🔂 dat_out_west[37:0] 🔂 stop_out[6:0] payload_1[20:0] payload_2[20:0] 🚡 payload_3[20:0] 🚡 payload_4[20:0] payload_5[20:0] 🔂 payload_6[20:0] 🚡 payload_7[20:0] 🔂 rec1 Leaf Filter: * -Ω 123 Show contents: In the signal list 📲 🔛 🕼 🚾 🚾 🗰 🗤 Filter: * ▼ 0 1 object selected Click on **Test** to see the different variables used in the simulation Click on 🗱 to visualize the signals

😣 🗐 🔲 🛛 Waveform 1 - SimV	'ision			
Eile Edit Yiew Explore Format Windows Help				
🚰 🍜 🗠 🗠 🕹 🛍	× [👞 🐏 [🏊]	₩Ę. \ \ ¶		Send To: 💽 🧱 🗈 🔐 😳 🎫 🚃
Search Names: Signal 💌	🗾 🛝 🛝 Sear	ch Times: Value 🔻	a a a	
x ₂ TimeA V = 0	🗲 📥 🕞			Time: 38 0 : 2000ns 🗹 🎧 = 🛛 🔐
× ⊕ 🔍 Baseline▼=0	Baseline =	= 0		
Name ▼	TimeA = 0 Cursor ▼ 0	500ns	1000ns	1500ns
📆 🗄 4/m dat_out_west[37:0]	No Value A⊧			
	No Value A⊧			
Payload_1[20:0]	No Value A⊧			
⊡	No Value A⊧			
H	No Value A⊧			
Here payload_4[20‡0]	No Value A⊧			
the payload_5t20;01	No Value Ab			
may pagroad_straves	No Value A⊧			
H. rec1	No Value A⊧			
tec2	No Value A⊧			
🚊 🖓 👘 rec3	No Value A⊧			
🕂 rec4	No Value A⊧			
🕂 🛱	No Value A⊧			
🕂 👘 rec6	No Value A⊧			
🗄 🕞 rec7	No Value A⊧			
······································	No Value A⊧			
🛨 👘 sent1	No Value A⊧			
terreta interreta interre	No Value A⊧			
tt	No Value A⊧			
±	No Value A⊧			
ento	No value H⊧			
the sent 7	No Value AN			
H-stop out[6:0]	No Value A⊧			
		100,000 200,000 300,000 400,000 15	00,000 600,000 700,000 800,0	00_ 900,000_ 1,000,0001,300,000ns >
10 J				0 objects selected
				0 05,0000 30100004

The waveform window appears Click on 💻 to fit the window with your signals

🛛 🗢 🗉 🔍 waveron	m 1 - Simvision		
Eile Edit View	Explore Format Window	3	Help
	·		🔐 . 🔔 Send To: 🟹 🚟 🕞 😎 🖼 🥅
] 67 (57 187 - 188 1	4 =0 =0 ∧ 10 ≤= 5	i <u>Plananananan</u>	😽 📲 🔤 📢 🖬 🔛
Search Names: Sig	nal 🕶 🔄 🍂	🛝 Search Tim 👔 👔 👔 👔 👔 👔	00
× 🕑 🔍 🛛 Baseli	ne▼=0	Baseline = 0 ())))))))))))))))))	00
Nome T	Current T		800.000mc 1.000.000mc 1.200.001
time dat_in_l	ocal[37:0] No Value A▶		
💆 🗄 🐨 🐻 dat_in_n	orth[37:0] No Value A⊧		000000000000000000000000000000000000000
🗦 🗄 🖓 dat_in_s	outh[37:0] No Value A⊧		00000000000000000000000000000000000000
🚽 🕀 🦏 dat_in_u	p[37:0] No Value A⊧		00000000000000000000000000000000000000
⊡dat_in_w	est[37:0] No Value A⊧	00000000000	0000000000000000000000000000000000000
H Gr dat_out_	down[37:0] No Value A⊧		
H dat_out_	eastL37:0J No Value HF		
H dat out	north[37:0] No Value Ak		
H dat out	south[37:0] No Value A▶		000000000000000000000000000000000000000
🛨 📲 dat_out_	up[37:0] No Value A⊧	ια απα απα απα από π	00000000000000000000000000000000000000
🗄 🖓 dat_out_	west[37:0] No Value A⊧		00000000000000000000000000000000000000
÷	No Value A⊧	00000000000	
🛨 🖓 👘 payload_	1[20:0] No Value A⊧	1000000000000000	00.000000000000000000000000000000000000
+ payload_	2[20:0] No Value A⊧		000000000000000000000000000000000000000
+ payload_	3[20:0] No Value A▶	AAAAAAAAAA	
H payload_	4[20:0] No Value A≯		
H payload	6[20:0] No Value AF		
H payload	7[20:0] No Value A		000000000000000000000000000000000000000
	No Value A		
	No Value A⊧		000000000000000000000000000000000000000
🛨 🐙 rec3	No Value A⊧		000000000000000000000000000000000000000
🛨 👘 rec4	No Value A▶		000000000000000000000000000000000000000
🕂 🖓 👘 rec5	No Value A⊧		000000000000000000000000000000000000000
🛨 🖓 rec6	No Value A⊧		000000000000000000000000000000000000000
⊞*[, rec7	No Value A⊧		
reset	No Value A⊧		
Hardin sent1	No Value HP		
H Sent3	No Value A	IO ob ject	
+ Sent4	No Value A⊧		
🕀 🖓 👘 sent5	No Value A⊧		
🛨 👫 sent6	No Value A⊧	000000000000000000000000000000000000000	00000000000000000000000000000000000000
🕀 🌾 sent7	No Value A▶	000000000000000000000000000000000000000))))))))))))))))))))))))))))))))))))))
E_ Ta. stop_out	[6:0] No Value A▶	(00	
		<u>المعامة المعامة المعام</u>	
10			0 objects selecter

From the waveform we can see that:

- The 100 flits that we sent arrived to their destinations
- No time violations are found, otherwise the signals will be red instead of green



Step 3: Power evaluation

- After we made sure that there are no time violations in our design, we proceed to evaluate the power consumption.
- The **dump.vcd** file contains the switching activities information of the test bench. We need to convert the **.vcd** file into **.saif** file.
- The **.saif** file will be used by Design Compiler Power Anlyzer to evaluate the power
- In your terminal and under ~/3D-ONoC/Post type the following command:



Step 3: Power evaluation

- For the evaluation, we execute a script that contains the necessary operations for power evaluation.
- The operations are almost the same as in the ones performed in Timing analysis (Step 1) of this post_layout simulation phase.
- The **power_LAXYZ.tcl** script file needed for this step is located in **./scripts**
- Next slides present **power_LAXYZ.tcl** file

Step 3: Power evaluation power_LAXYZ.tcl (1/2)

Define the variable which we will use

"router_LAXYZ"
"router_LAXYZ_final.vnet
"router_LAXYZ.spef"
"router_LAXYZ.sdf"
"router_LAXYZ.sdc"
"router_LAXYZ.saif"

Step 1: Set the libraries:
set target_library "~/lib/typical.db"
set synthetic_library "~/lib/dw_foundation.sldb"
set link_library [concat "*" \$target_library \$synthetic_library]
set symbol_library ""~/lib/generic.sdb"
define_design_lib WORK -path ./WORK # redirect the log files to a new folder "WORK"

Step 3: Power evaluation power_LAXYZ.tcl (1/2)

Step 2: Read post layout netlist#### read file -format verilog ./input/\$vnet file current_design \$base_name link #### Delay and RC information#### read_sdc ./input/\$sdc_file read sdf./input/\$sdf file read parasitics ./input/\$spef file #### Read switching activities information#### reset switching activity read saif -input \$saif file -instance top/dut -unit ns -scale 1 #### Generate reports#### report timing > ./reports/timing report \${base name}.txt report reference -hier > ./reports/reference report \${base name}.txt report power -hier > ./reports/power report \${base name}.txt



Step 3: Power evaluation Execute the script

😣 🗈 Execute Script File	
Look in: 🔄 /home/zxp035/3D_ONoC/Post/scripts/) 💣 🔝 🏛
<pre> power_LAXYZ.tcl sta_LAXYZ.tcl </pre>	
File <u>n</u> ame: power_LAXYZ.tcl	<u>O</u> pen
File type: Script Files (*.script *.scr *.dcs *.dcv *.dc *.dcfpga *.con *.tcl *.tcl)	Cancel
Echo commands	
└ <u>V</u> erbose	

To run the TCL script, click File> Execute script in Design Compiler Go to ./scripts, select power_LAXYZ.tcl and click Open

Step 3: Power evaluation Reports

Hierarchy Power Power Power Power Power Power Power No router_LAXYZ 4.395 7.763 2.10e+05 222.387 100.0 cbar (crossbar_MOUT7_NIN7_WIDTH34) 3.87e-02 0.361 3.52e+04 35.554 16.0 output_loop[6].cbar_mux (mux_out_n[7/WIDTH34_0) 5.147 2.3 output_loop[5].cbar_mux (mux_out_nin7_WIDTH34_1) 2.87e+03 1.83e+03 4.09e+03 4.093 1.8 output_loop[4].cbar_mux (mux_out_nin7_WIDTH34_2) 2.26e+03 2.36e+03 3.742 1.7 output_loop[3].cbar_mux (mux_out_nin7_WIDTH34_3) 3.742 1.7 0 output_loop[2].cbar_mux (mux_out_nin7_WIDTH34_4) 3.82e+03 3.820 1.7 output_loop[1].cbar_mux (mux_out_nin7_WIDTH34_5) 3.82e+03 3.820 1.7 output_loop[0].cbar_mux (mux_out_nin7_WIDTH34_5) 3.82e+03 4.396 2.0 output_loop[0].cbar_mux (mux_out_nin7_WIDTH34_5) 3.41e+03 4.113 1.8 <t< th=""><th></th><th></th><th>Switch</th><th>Int</th><th>Leak</th><th>Total</th><th></th></t<>			Switch	Int	Leak	Total	
router_LAXYZ 4.395 7.763 2.10e+05 222.387 100.0 cbar (crossbar_NOUT7_NIN7_WIDTH34) 3.87e-02 0.361 3.52e+04 35.554 16.0 output_loop[6].cbar_mux (mux_out_nin7_WIDTH34_0) 1.15e-02 2.35e-03 5.13e+03 5.147 2.3 output_loop[5].cbar_mux (mux_out_nin7_WIDTH34_1) 2.87e-03 1.83e-03 4.09e+03 4.093 1.8 output_loop[4].cbar_mux (mux_out_nin7_WIDTH34_2) 2.26e-03 2.36e-03 3.74e+03 3.742 1.7 output_loop[3].cbar_mux (mux_out_nin7_WIDTH34_3) 1.14e-02 2.42e-03 4.34e+03 4.356 2.0 output_loop[2].cbar_mux (mux_out_nin7_WIDTH34_4) 2.55e-03 1.96e-03 3.82e+03 3.820 1.7 output_loop[1].cbar_mux (mux_out_nin7_WIDTH34_5) 2.41e-03 2.33e-03 4.39e+03 4.396 2.0 output_loop[0].cbar_mux (mux_out_nin7_WIDTH34_5) 2.41e-03 2.33e-03 4.39e+03 4.396 2.0 output_loop[0].cbar_mux (mux_out_nin7_WIDTH34_6) 4.113 1.8 sw_allc (sw_alloc_NOUT7)		Hierarchy	Power	Power	Power	Power	%
router_LAXYZ 4.395 7.763 2.10e+05 222.387 100.0 cbar (crossbar_NOUT7_NIN7_WIDTH34) 3.87e-02 0.361 3.52e+04 35.554 16.0 output_loop[6].cbar_mux (mux_out_nin7_WIDTH34_0) 1.15e-02 2.35e-03 5.13e+03 5.147 2.3 output_loop[5].cbar_mux (mux_out_nin7_WIDTH34_1) 2.87e-03 1.88e-03 4.09e+03 4.093 1.8 output_loop[4].cbar_mux (mux_out_nin7_WIDTH34_2) 2.26e-03 2.36e-03 3.74e+03 3.742 1.7 output_loop[3].cbar_mux (mux_out_nin7_WIDTH34_3) 1.14e-02 2.42e-03 4.34e+03 4.356 2.0 output_loop[2].cbar_mux (mux_out_nin7_WIDTH34_4) 2.55e-03 1.96e-03 3.82e+03 3.820 1.7 output_loop[1].cbar_mux (mux_out_nin7_WIDTH34_6) 2.41e-03 2.38e-03 4.39e+03 4.396 2.0 output_loop[0].cbar_mux (mux_out_nin7_WIDTH34_6) 2.41e-03 2.31e-03 4.39e+03 4.396 2.0 output_loop[0].cbar_mux (mux_out_nin7_WIDTH34_6) 1.82e-03 1.79e-03 4.1113 1.8 sw_all	٢						
cbar (crossbar_NOUT7_NIN7_WIDTH34) 3.87e-02 0.361 3.52e+04 35.554 16.0 output_loop[6].cbar_mux (mux_out_nin7_WIDTH34_0) 1.15e-02 2.35e-03 5.13e+03 5.147 2.3 output_loop[5].cbar_mux (mux_out_nin7_WIDTH34_1) 2.87e-03 1.88e-03 4.09e+03 4.093 1.8 output_loop[4].cbar_mux (mux_out_nin7_WIDTH34_2) 2.26e-03 2.36e-03 3.74e+03 3.742 1.7 output_loop[3].cbar_mux (mux_out_nin7_WIDTH34_3) 1.14e-02 2.42e-03 4.34e+03 4.356 2.0 output_loop[1].cbar_mux (mux_out_nin7_WIDTH34_4) 2.55e-03 1.96e-03 3.82e+03 3.820 1.7 output_loop[1].cbar_mux (mux_out_nin7_WIDTH34_4) 2.55e-03 1.96e-03 3.82e+03 4.396 2.0 output_loop[0].cbar_mux (mux_out_nin7_WIDTH34_6) 2.41e-03 2.38e-03 4.39e+03 4.396 2.0 output_loop[0].cbar_mux (mux_out_nin7_WIDTH34_6) 1.82e-03 1.79e-03 4.11e+03 4.113 1.8 sw_allc (sw_alloc_NOUT7) 8.37e-04 1.446 3.62e+04 37.677 16.9 01[6].spg (stop_go_0) 2.94e+06 1.39e+02 419.688 0.434	L	router_LAXYZ	4.395	7.763	2.10e+05	222.387	100.0
output_loop[6].cbar_mux (mux_out_n_in7_WIDTH34_0) 1.15e-02 2.35e-03 5.13e+03 5.147 2.3 output_loop[5].cbar_mux (mux_out_n_in7_WIDTH34_1) 2.87e-03 1.83e-03 4.09e+03 4.093 1.8 output_loop[4].cbar_mux (mux_out_n_in7_WIDTH34_2) 2.26e-03 2.36e-03 3.74e+03 3.742 1.7 output_loop[3].cbar_mux (mux_out_n_in7_WIDTH34_3) 1.14e-02 2.42e-03 4.34e+03 4.356 2.0 output_loop[2].cbar_mux (mux_out_n_in7_WIDTH34_4) 2.55e-03 1.96e-03 3.82e+03 3.820 1.7 output_loop[1].cbar_mux (mux_out_n_in7_WIDTH34_5) 2.41e-03 2.33e-03 4.39e+03 4.396 2.0 output_loop[0].cbar_mux (mux_out_n_in7_WIDTH34_6) 1.82e-03 1.79e-03 4.11e+03 4.113 1.8 sw_allc (sw_alloc_NOUT7) 8.37e-04 1.446 3.62e+04 37.677 16.9 ol[6].spg (stop_go_0) 2.94e-06 1.39e-02 419.688 0.434 0.2 ol[6].mat_arb (matrix_arb_formultistage_SIZE7_0) 5.22e-05 0.145 3.35e+03 3.423 1.5 ol[4].spg (stop_go_2) 3.13e-06 1.39e-02 419.688 0.434 0.2 ol[4].mat_arb (matrix_arb_formultistage_SIZE7_2)		cbar (crossbar_NOUT7_NIN7_WIDTH34)	3.87e-02	0.361	3.52e+04	35.554	16.0
1.15e-02 2.35e-03 5.13e+03 5.147 2.3 output_loop[5].cbar_mux (mux_out_n_in7_WIDTH34_1) 2.87e-03 1.83e-03 4.09e+03 4.093 1.8 output_loop[4].cbar_mux (mux_out_n_in7_WIDTH34_2) 2.26e-03 2.36e-03 3.74e+03 3.742 1.7 output_loop[3].cbar_mux (mux_out_n_in7_WIDTH34_3) 1.14e-02 2.42e+03 4.34e+03 4.356 2.0 output_loop[2].cbar_mux (mux_out_n_in7_WIDTH34_4) 2.55e+03 1.96e+03 3.82e+03 3.820 1.7 output_loop[1].cbar_mux (mux_out_n_in7_WIDTH34_5) 2.41e+03 2.33e+03 4.39e+03 4.396 2.0 output_loop[0].cbar_mux (mux_out_n_in7_WIDTH34_6) 1.82e+03 1.79e+03 4.11e+03 4.113 1.8 sw_allc (sw_alloc_NOUT7) 8.37e+04 1.446 3.62e+04 37.677 16.9 ol[6].spg (stop_go_0) 2.94e+06 1.39e+02 419.688 0.434 0.2 ol[6].mat_arb (matrix_arb_formultistage_SIZE7_1) 5.05e+05 0.145 3.28e+03 3.423 1.5 ol[4].spg (stop_go_2) 3.13e+06 1.39e+02 419.688 0.434 0.2 ol[4].mat_arb (matrix_arb_formultistage_SIZE7_2)		output_loop[6].cbar_mux (mux_out_n	_in7_WIDT	H34_0)			
output_loop[5].cbar_mux (mux_out_n_in7_WIDTH34_1) 2.87e-03 1.83e-03 4.09e+03 4.093 1.8 output_loop[4].cbar_mux (mux_out_n_in7_WIDTH34_2) 2.26e-03 2.36e-03 3.74e+03 3.742 1.7 output_loop[3].cbar_mux (mux_out_n_in7_WIDTH34_3) 1.14e-02 2.42e-03 4.34e+03 4.356 2.0 output_loop[2].cbar_mux (mux_out_n_in7_WIDTH34_4) 2.55e-03 1.96e-03 3.82e+03 3.820 1.7 output_loop[1].cbar_mux (mux_out_n_in7_WIDTH34_5) 2.41e-03 2.33e-03 4.39e+03 4.396 2.0 output_loop[0].cbar_mux (mux_out_n_in7_WIDTH34_6) 1.82e-03 1.79e-03 4.11e+03 4.113 1.8 sw_allc (sw_alloc_NOUT7) 8.37e-04 1.446 3.62e+04 37.677 16.9 ol[6].spg (stop_go_0) 2.94e-06 1.39e-02 419.688 0.434 0.2 ol[6].mat_arb (matrix_arb_formultistage_SIZE7_0) 5.22e-05 0.145 3.35e+03 3.496 1.6 ol[5].spg (stop_go_1) 4.27e-06 1.39e-02 476.288 0.490 0.2 ol[5].mat_arb (matrix_arb_formultistage_SIZE7_1) 5.05e-05 0.145 3.28e+03 3.423 1.5 ol[4].spg (stop_go_2) 3.13e-06 1.39e-02 419.688 0.434 0.2 ol[4].mat_arb (matrix_arb_formultistage_SIZE7_2)			1.15e-02	2.35e-03	5.13e+03	5.147	2.3
2.87e-03 1.83e-03 4.09e+03 4.093 1.8 output_loop[4].cbar_mux (mux_out_n_in7_WIDTH34_2) 2.26e-03 2.36e-03 3.74e+03 3.742 1.7 output_loop[3].cbar_mux (mux_out_n_in7_WIDTH34_3) 1.14e-02 2.42e-03 4.34e+03 4.356 2.0 output_loop[2].cbar_mux (mux_out_n_in7_WIDTH34_4) 2.55e-03 1.96e-03 3.82e+03 3.820 1.7 output_loop[1].cbar_mux (mux_out_n_in7_WIDTH34_5) 2.41e-03 2.33e-03 4.39e+03 4.396 2.0 output_loop[0].cbar_mux (mux_out_n_in7_WIDTH34_6) 1.82e-03 1.79e-03 4.11e+03 4.113 1.8 sw_allc (sw_alloc_NOUT7) 8.37e-04 1.446 3.62e+04 37.677 16.9 ol[6].spg (stop_go_0) 2.94e-06 1.39e-02 419.688 0.434 0.2 ol[6].mat_arb (matrix_arb_formultistage_SIZE7_0) 5.22e-05 0.145 3.35e+03 3.496 1.6 ol[5].spg (stop_go_1) 4.27e-06 1.39e-02 476.288 0.490 0.2 ol[5].mat_arb (matrix_arb_formultistage_SIZE7_1) 5.05e-05 0.145 3.28e+03 3.423 1.5 ol[4].spg (stop_go_2) 3.13e-06 1.39e-02 419.688 0.434 0.2 ol[4].mat_arb (matrix_arb_formultistage_SIZE7_2)		output_loop[5].cbar_mux (mux_out_n	_in7_WIDT	H34_1)			
output_loop[4].cbar_mux (mux_out_n_in7_WIDTH34_2) 2.26e-03 2.36e-03 3.74e+03 3.742 1.7 output_loop[3].cbar_mux (mux_out_n_in7_WIDTH34_3) 1.14e-02 2.42e-03 4.34e+03 4.356 2.0 output_loop[2].cbar_mux (mux_out_n_in7_WIDTH34_4) 2.55e-03 1.96e-03 3.82e+03 3.820 1.7 output_loop[1].cbar_mux (mux_out_n_in7_WIDTH34_5) 2.41e-03 2.33e-03 4.39e+03 4.396 2.0 output_loop[0].cbar_mux (mux_out_n_in7_WIDTH34_6) 1.82e-03 1.79e-03 4.11e+03 4.113 1.8 sw_allc (sw_alloc_NOUT7) 8.37e-04 1.446 3.62e+04 37.677 16.9 ol[6].spg (stop_go_0) 2.94e-06 1.39e-02 419.688 0.434 0.2 ol[6].mat_arb (matrix_arb_formultistage_SIZE7_0) 5.22e-05 0.145 3.35e+03 3.496 1.6 ol[5].spg (stop_go_1) 4.27e-06 1.39e-02 476.288 0.490 0.2 ol[5].mat_arb (matrix_arb_formultistage_SIZE7_1) 5.05e-05 0.145 3.28e+03 3.423 1.5 ol[4].spg (stop_go_2) 3.13e-06 1.39e-02 419.688 0.434 0.2 ol[4].mat_arb (matrix_arb_formultistage_SIZE7_2)			2.87e-03	1.83e-03	4.09e+03	4.093	1.8
2.26e-03 2.36e-03 3.74e+03 3.742 1.7 output_loop[3].cbar_mux (mux_out_n_in7_WIDTH34_3) 1.14e-02 2.42e-03 4.34e+03 4.356 2.0 output_loop[2].cbar_mux (mux_out_n_in7_WIDTH34_4) 2.55e-03 1.96e-03 3.82e+03 3.820 1.7 output_loop[1].cbar_mux (mux_out_n_in7_WIDTH34_5) 2.41e-03 2.33e-03 4.39e+03 4.396 2.0 output_loop[0].cbar_mux (mux_out_n_in7_WIDTH34_6) 1.82e-03 1.79e-03 4.11e+03 4.113 1.8 sw_allc (sw_alloc_NOUT7) 8.37e-04 1.446 3.62e+04 37.677 16.9 ol[6].spg (stop_go_0) 2.94e-06 1.39e-02 419.688 0.434 0.2 ol[6].mat_arb (matrix_arb_formultistage_SIZE7_0) 5.22e-05 0.145 3.35e+03 3.496 1.6 ol[5].spg (stop_go_1) 4.27e-06 1.39e-02 476.288 0.490 0.2 ol[5].mat_arb (matrix_arb_formultistage_SIZE7_1) 5.05e-05 0.145 3.28e+03 3.423 1.5 ol[4].spg (stop_go_2) 3.13e-06 1.39e-02 419.688 0.434 0.2 ol[4].mat_arb (matrix_arb_formultistage_SIZE7_2)		output_loop[4].cbar_mux (mux_out_n	_in7_WIDT	H34_2)			
output_loop[3].cbar_mux (mux_out_n_in7_WIDTH34_3) 1.14e-02 2.42e-03 4.34e+03 4.356 2.0 output_loop[2].cbar_mux (mux_out_n_in7_WIDTH34_4) 2.55e-03 1.96e-03 3.82e+03 3.820 1.7 output_loop[1].cbar_mux (mux_out_n_in7_WIDTH34_5) 2.41e-03 2.33e-03 4.39e+03 4.396 2.0 output_loop[0].cbar_mux (mux_out_n_in7_WIDTH34_6) 1.82e-03 1.79e-03 4.11e+03 4.113 1.8 sw_allc (sw_alloc_NOUT7) 8.37e-04 1.446 3.62e+04 37.677 16.9 ol[6].spg (stop_go_0) 2.94e-06 1.39e-02 419.688 0.434 0.2 ol[6].mat_arb (matrix_arb_formultistage_SIZE7_0) 5.22e-05 0.145 3.35e+03 3.496 1.6 ol[5].spg (stop_go_1) 4.27e-06 1.39e-02 476.288 0.490 0.2 ol[5].mat_arb (matrix_arb_formultistage_SIZE7_1) 5.05e-05 0.145 3.28e+03 3.423 1.5 ol[4].spg (stop_go_2) 3.13e-06 1.39e-02 419.688 0.434 0.2 ol[4].mat_arb (matrix_arb_formultistage_SIZE7_2) 0.145 3.28e+03 3.423 1.5			2.26e-03	2.36e-03	3.74e+03	3.742	1.7
1.14e-02 2.42e-03 4.34e+03 4.356 2.0 output_loop[2].cbar_mux (mux_out_n_in7_WIDTH34_4) 2.55e-03 1.96e-03 3.82e+03 3.820 1.7 output_loop[1].cbar_mux (mux_out_n_in7_WIDTH34_5) 2.41e-03 2.33e-03 4.39e+03 4.396 2.0 output_loop[0].cbar_mux (mux_out_n_in7_WIDTH34_6) 1.82e-03 1.79e-03 4.11e+03 4.113 1.8 sw_allc (sw_alloc_NOUT7) 8.37e-04 1.446 3.62e+04 37.677 16.9 ol[6].spg (stop_go_0) 2.94e-06 1.39e-02 419.688 0.434 0.2 ol[6].mat_arb (matrix_arb_formultistage_SIZE7_0) 5.22e-05 0.145 3.35e+03 3.496 1.6 ol[5].spg (stop_go_1) 4.27e-06 1.39e-02 476.288 0.490 0.2 ol[5].mat_arb (matrix_arb_formultistage_SIZE7_1) 5.05e-05 0.145 3.28e+03 3.423 1.5 ol[4].spg (stop_go_2) 3.13e-06 1.39e-02 419.688 0.434 0.2 ol[4].mat_arb (matrix_arb_formultistage_SIZE7_2)		output_loop[3].cbar_mux (mux_out_n	_in7_WIDT	H34_3)			
output_loop[2].cbar_mux (mux_out_n_in7_WIDTH34_4) 2.55e-03 1.96e-03 3.82e+03 3.820 1.7 output_loop[1].cbar_mux (mux_out_n_in7_WIDTH34_5) 2.41e-03 2.33e-03 4.39e+03 4.396 2.0 output_loop[0].cbar_mux (mux_out_n_in7_WIDTH34_6) 1.82e-03 1.79e-03 4.11e+03 4.113 1.8 sw_allc (sw_alloc_NOUT7) 8.37e-04 1.446 3.62e+04 37.677 16.9 ol[6].spg (stop_go_0) 2.94e-06 1.39e-02 419.688 0.434 0.2 ol[6].mat_arb (matrix_arb_formultistage_SIZE7_0) 5.22e-05 0.145 3.35e+03 3.496 1.6 ol[5].spg (stop_go_1) 4.27e-06 1.39e-02 476.288 0.490 0.2 ol[5].mat_arb (matrix_arb_formultistage_SIZE7_1) 5.05e-05 0.145 3.28e+03 3.423 1.5 ol[4].spg (stop_go_2) 3.13e-06 1.39e-02 419.688 0.434 0.2 ol[4].mat_arb (matrix_arb_formultistage_SIZE7_2) 3.13e-06 1.39e-02 419.688 0.434 0.2			1.14e-02	2.42e-03	4.34e+03	4.356	2.0
2.55e-03 1.96e-03 3.82e+03 3.820 1.7 output_loop[1].cbar_mux (mux_out_n_in7_WIDTH34_5) 2.41e-03 2.33e-03 4.39e+03 4.396 2.0 output_loop[0].cbar_mux (mux_out_n_in7_WIDTH34_6) 1.82e-03 1.79e-03 4.11e+03 4.113 1.8 sw_allc (sw_alloc_NOUT7) 8.37e-04 1.446 3.62e+04 37.677 16.9 ol[6].spg (stop_go_0) 2.94e-06 1.39e-02 419.688 0.434 0.2 ol[6].mat_arb (matrix_arb_formultistage_SIZE7_0) 5.22e-05 0.145 3.35e+03 3.496 1.6 ol[5].spg (stop_go_1) 4.27e-06 1.39e-02 476.288 0.490 0.2 ol[5].mat_arb (matrix_arb_formultistage_SIZE7_1) 5.05e-05 0.145 3.28e+03 3.423 1.5 ol[4].spg (stop_go_2) 3.13e-06 1.39e-02 419.688 0.434 0.2 ol[4].mat_arb (matrix_arb_formultistage_SIZE7_2)		output_loop[2].cbar_mux (mux_out_n	_in7_WIDT	H34_4)			
output_loop[1].cbar_mux (mux_out_n_in7_WIDTH34_5) 2.41e-03 2.33e-03 4.39e+03 4.396 2.0 output_loop[0].cbar_mux (mux_out_n_in7_WIDTH34_6) 1.82e-03 1.79e-03 4.11e+03 4.113 1.8 sw_allc (sw_alloc_NOUT7) 8.37e-04 1.446 3.62e+04 37.677 16.9 ol[6].spg (stop_go_0) 2.94e-06 1.39e-02 419.688 0.434 0.2 ol[6].mat_arb (matrix_arb_formultistage_SIZE7_0) 5.22e-05 0.145 3.35e+03 3.496 1.6 ol[5].spg (stop_go_1) 4.27e-06 1.39e-02 476.288 0.490 0.2 ol[5].mat_arb (matrix_arb_formultistage_SIZE7_1) 5.05e-05 0.145 3.28e+03 3.423 1.5 ol[4].spg (stop_go_2) 3.13e-06 1.39e-02 419.688 0.434 0.2 ol[4].mat_arb (matrix_arb_formultistage_SIZE7_2) 3.13e-06 1.39e-02 419.688 0.434 0.2			2.55e-03	1.96e-03	3.82e+03	3.820	1.7
2.41e-03 2.33e-03 4.39e+03 4.396 2.0 output_loop[0].cbar_mux (mux_out_n_in7_WIDTH34_6) 1.82e-03 1.79e-03 4.11e+03 4.113 1.8 sw_allc (sw_alloc_NOUT7) 8.37e-04 1.446 3.62e+04 37.677 16.9 ol[6].spg (stop_go_0) 2.94e-06 1.39e-02 419.688 0.434 0.2 ol[6].mat_arb (matrix_arb_formultistage_SIZE7_0) 5.22e-05 0.145 3.35e+03 3.496 1.6 ol[5].spg (stop_go_1) 4.27e-06 1.39e-02 476.288 0.490 0.2 ol[5].mat_arb (matrix_arb_formultistage_SIZE7_1) 5.05e-05 0.145 3.28e+03 3.423 1.5 ol[4].spg (stop_go_2) 3.13e-06 1.39e-02 419.688 0.434 0.2 ol[4].mat_arb (matrix_arb_formultistage_SIZE7_2)		output_loop[1].cbar_mux (mux_out_n	_in7_WIDT	H34_5)			
output_loop[0].cbar_mux (mux_out_n_in7_WIDTH34_6) 1.82e-03 1.79e-03 4.11e+03 4.113 1.8 sw_allc (sw_alloc_NOUT7) 8.37e-04 1.446 3.62e+04 37.677 16.9 ol[6].spg (stop_go_0) 2.94e-06 1.39e-02 419.688 0.434 0.2 ol[6].mat_arb (matrix_arb_formultistage_SIZE7_0) 5.22e-05 0.145 3.35e+03 3.496 1.6 ol[5].spg (stop_go_1) 4.27e-06 1.39e-02 476.288 0.490 0.2 ol[5].mat_arb (matrix_arb_formultistage_SIZE7_1) 5.05e-05 0.145 3.28e+03 3.423 1.5 ol[4].spg (stop_go_2) 3.13e-06 1.39e-02 419.688 0.434 0.2 ol[4].mat_arb (matrix_arb_formultistage_SIZE7_2) 3.12e-06 1.39e-02 419.688 0.434 0.2			2.41e-03	2.33e-03	4.39e+03	4.396	2.0
1.82e-03 1.79e-03 4.11e+03 4.113 1.8 sw_allc (sw_alloc_NOUT7) 8.37e-04 1.446 3.62e+04 37.677 16.9 ol[6].spg (stop_go_0) 2.94e-06 1.39e-02 419.688 0.434 0.2 ol[6].mat_arb (matrix_arb_formultistage_SIZE7_0) 5.22e-05 0.145 3.35e+03 3.496 1.6 ol[5].spg (stop_go_1) 4.27e-06 1.39e-02 476.288 0.490 0.2 ol[5].mat_arb (matrix_arb_formultistage_SIZE7_1) 5.05e-05 0.145 3.28e+03 3.423 1.5 ol[4].spg (stop_go_2) 3.13e-06 1.39e-02 419.688 0.434 0.2 ol[4].mat_arb (matrix_arb_formultistage_SIZE7_2) 0.145 3.28e+03 3.423 1.5		output_loop[0].cbar_mux (mux_out_n	_in7_WIDT	H34_6)			
sw_allc (sw_alloc_NOUT7) 8.37e-04 1.446 3.62e+04 37.677 16.9 ol[6].spg (stop_go_0) 2.94e-06 1.39e-02 419.688 0.434 0.2 ol[6].mat_arb (matrix_arb_formultistage_SIZE7_0) 5.22e-05 0.145 3.35e+03 3.496 1.6 ol[5].spg (stop_go_1) 4.27e-06 1.39e-02 476.288 0.490 0.2 ol[5].mat_arb (matrix_arb_formultistage_SIZE7_1) 5.05e-05 0.145 3.28e+03 3.423 1.5 ol[4].spg (stop_go_2) 3.13e-06 1.39e-02 419.688 0.434 0.2 ol[4].mat_arb (matrix_arb_formultistage_SIZE7_2) 0.145 3.28e+03 3.423 0.5			1.82e-03	1.79e-03	4.11e+03	4.113	1.8
ol[6].spg (stop_go_0) 2.94e-06 1.39e-02 419.688 0.434 0.2 ol[6].mat_arb (matrix_arb_formultistage_SIZE7_0) 5.22e-05 0.145 3.35e+03 3.496 1.6 ol[5].spg (stop_go_1) 4.27e-06 1.39e-02 476.288 0.490 0.2 ol[5].mat_arb (matrix_arb_formultistage_SIZE7_1) 5.05e-05 0.145 3.28e+03 3.423 1.5 ol[4].spg (stop_go_2) 3.13e-06 1.39e-02 419.688 0.434 0.2 ol[4].mat_arb (matrix_arb_formultistage_SIZE7_2)		<pre>sw_allc (sw_alloc_NOUT7)</pre>	8.37e-04	1.446	3.62e+04	37.677	16.9
ol[6].mat_arb (matrix_arb_formultistage_SIZE7_0) 5.22e-05 0.145 3.35e+03 3.496 1.6 ol[5].spg (stop_go_1) 4.27e-06 1.39e-02 476.288 0.490 0.2 ol[5].mat_arb (matrix_arb_formultistage_SIZE7_1) 5.05e-05 0.145 3.28e+03 3.423 1.5 ol[4].spg (stop_go_2) 3.13e-06 1.39e-02 419.688 0.434 0.2 ol[4].mat_arb (matrix_arb_formultistage_SIZE7_2)		ol[6].spg (stop_go_0)	2.94e-06	1.39e-02	419.688	0.434	0.2
5.22e-05 0.145 3.35e+03 3.496 1.6 ol[5].spg (stop_go_1) 4.27e-06 1.39e-02 476.288 0.490 0.2 ol[5].mat_arb (matrix_arb_formultistage_SIZE7_1) 5.05e-05 0.145 3.28e+03 3.423 1.5 ol[4].spg (stop_go_2) 3.13e-06 1.39e-02 419.688 0.434 0.2 ol[4].mat_arb (matrix_arb_formultistage_SIZE7_2)		ol[6].mat_arb (matrix_arb_formultistage_SIZE7_0)					
ol[5].spg (stop_go_1) 4.27e-06 1.39e-02 476.288 0.490 0.2 ol[5].mat_arb (matrix_arb_formultistage_SIZE7_1) 5.05e-05 0.145 3.28e+03 3.423 1.5 ol[4].spg (stop_go_2) 3.13e-06 1.39e-02 419.688 0.434 0.2 ol[4].mat_arb (matrix_arb_formultistage_SIZE7_2)			5.22e-05	0.145	3.35e+03	3.496	1.6
ol[5].mat_arb (matrix_arb_formultistage_SIZE7_1) 5.05e-05 0.145 3.28e+03 3.423 1.5 ol[4].spg (stop_go_2) 3.13e-06 1.39e-02 419.688 0.434 0.2 ol[4].mat_arb (matrix_arb_formultistage_SIZE7_2)		ol[5].spg (stop_go_1)	4.27e-06	1.39e-02	476.288	0.490	0.2
5.05e-05 0.145 3.28e+03 3.423 1.5 ol[4].spg (stop_go_2) 3.13e-06 1.39e-02 419.688 0.434 0.2 ol[4].mat_arb (matrix_arb_formultistage_SIZE7_2)		ol[5].mat_arb (matrix_arb_formulti:	stage_SIZ	E7_1)			
ol[4].spg (stop_go_2) 3.13e-06 1.39e-02 419.688 0.434 0.2 ol[4].mat_arb (matrix_arb_formultistage_SIZE7_2)			5.05e-05	0.145	3.28e+03	3.423	1.5
ol[4].mat_arb (matrix_arb_formultistage_SIZE7_2)		ol[4].spg (stop_go_2)	3.13e-06	1.39e-02	419.688	0.434	0.2
		ol[4].mat_arb (matrix_arb_formulti:	stage_SIZ	E7_2)			
4.25e-05 0.145 3.21e+03 3.356 1.5			4.25e-05	0.145	3.21e+03	3.356	1.5
ol[3].spg (stop_go_3) 3.46e-06 1.39e-02 419.688 0.434 0.2		ol[3].spg (stop_go_3)	3.46e-06	1.39e-02	419.688	0.434	0.2
ol[3].mat_arb (matrix_arb_formultistage_SIZE7_3)		ol[3].mat_arb (matrix_arb_formulti:	stage_SIZ	E7_3)			
Log History	L	og History					

design_vision>

The total power consumption of the 3D-ONoC router is 222.387 uW:

- Leakage (static): 210 uW
- Internal (dynamic): 7.763 uW

156

Switching (net): 4.395 uW

A detailed power reports is shown in dc_shell console.

The report shows the static, dynamic and switching power consumption per module



<== Back to Contents

5. Pad Insertion



Objectives

- After completing this tutorial you will be able to:
 - Reperform the Place and Route phase while inserting the Input/Output (IO) pads
 - Establish the connection between the IO pins/ IO pads, and the input signals of 3D-ONoC router
 - Generate the final netlist and other output files
- This tutorial is performed is based on modifying input-files and creating a TCL script



Contents

- Requirements
- Pad Insertion directory structure
- Environment
- Step 1: Modify router_LAXYZ.vnet
- Step 2: Make router_LAXYZ.io
- Step 3: Make lopad_LAXYZ.tcl script
- Step 4: Script execution
- Final layout



Requirements

- Before starting the post-layout, you should have already finished the four previous phases:
 - Design Synthesis (DS)
 - Place & Route (P&R)
 - Design Check (LVS and DCR)
 - Post-layout simulations
- We should create a new directory: ~/3D-ONoC/Iopad where the IO pad insertion is performed.



Post-layout directory structure





Environment

\$ tcsh
/home/zxp035/3D-ONoC% cd Iopad/
/home/zxp035/3D-ONoC/Pad%

Make sure that you are working under **cshr** environment. Otherwise type **tcsh**. Go to */home/zxp035/3D-ONoC/lopad* where the Pad folder is located



Environment

% cp ../Synthesis/output_files/router_LAXYZ.vnet ./input % cp ../Synthesis/output_files/router_LAXYZ.sdc ./input

First, we need t copy the *router_LAXYZ.vnet* and *router_LAXYZ.sdc* files generated from the synthesis phase which will be used as input for the Pad insertion phase. Type: % cp ../Synthesis/ouput_files/router_LAXYZ.vnet ./input % cp ../Synthesis/ouput_files/router_LAXYZ.sdc ./input

Step 1: Modify router_LAXYZ.vnet (1/6)

module router_LAXYZ (clk_pad, reset_pad, data_in_pad, data_out_pad, stop_in_pad, stop_out_pad, xaddr_pad,

yaddr_pad, zaddr_pad);

input [265:0] data_in_pad;

output [265:0] data_out_pad;

input [6:0] stop_in_pad;

output [6:0] stop_out_pad;

input [2:0] xaddr pad;

input [2:0] yaddr_pad;

input [2:0] zaddr_pad;

input clk_pad, reset_pad;

wire [265:0] data_in; wire [265:0] data_out; wire [6:0] stop_in; wire [6:0] stop_out; wire [2:0] xaddr; wire [2:0] yaddr; wire [2:0] zaddr;

Step 1: Modify router_LAXYZ.vnet (2/6)

```
wire [265:0] cbar_data_in;
wire [6:0] sw_req;
wire [48:0] port_req;
wire [6:0] sw_grant;
wire [6:0] data_sent;
wire [48:0] sw cntrl;
CORNER PAD corner 0 inst ();
IN PAD data in 0 inst (.PAD(data in pad[0]), .O(data in[0]));
IN PAD data in 1 inst (.PAD(data in pad[1]), .O(data in[1]));
// ...
// Repeat
// ...
IN PAD data in 137 inst (.PAD(data in pad[137]), .O(data in[137]));
IN PAD data in 138 inst (.PAD(data in pad[138]), .O(data in[138]));
VDD PAD vdd 0 inst ();
VSS PAD vss 0 inst ();
```

wire clk, reset;

wire n1, n2, n3, n4, n5, n6, n7;

Step 1: Modify router_LAXYZ.vnet (3/6)

CORNER_PAD corner_1_inst ();

// ...

// Repeat

// ...

IN_PAD data_in_264_inst (.PAD(data_in_pad[264]), .O(data_in[264])); IN_PAD data_in_265_inst (.PAD(data_in_pad[265]), .O(data_in[265])); OUT_PAD data_out_0_inst (.PAD(data_out_pad[0]), .I(data_out[0])); OUT_PAD data_out_1_inst (.PAD(data_out_pad[1]), .I(data_out[1])); OUT_PAD data_out_2_inst (.PAD(data_out_pad[2]), .I(data_out[2])); OUT_PAD data_out_3_inst (.PAD(data_out_pad[3]), .I(data_out[3])); OUT_PAD data_out_4_inst (.PAD(data_out_pad[3]), .I(data_out[3])); OUT_PAD data_out_5_inst (.PAD(data_out_pad[4]), .I(data_out[4])); OUT_PAD data_out_5_inst (.PAD(data_out_pad[5]), .I(data_out[5])); OUT_PAD data_out_6_inst (.PAD(data_out_pad[6]), .I(data_out[6])); OUT_PAD data_out_7_inst (.PAD(data_out_pad[7]), .I(data_out[7])); OUT_PAD data_out_8_inst (.PAD(data_out_pad[8]), .I(data_out[8])); OUT_PAD data_out_9_inst (.PAD(data_out_pad[9]), .I(data_out[9])); OUT_PAD data_out_10_inst (.PAD(data_out_pad[10]), .I(data_out[10])); OUT_PAD data_out_11_inst (.PAD(data_out_pad[11]), .I(data_out[11]));

Step 1: Modify router_LAXYZ.vnet (4/6)

```
VDD_PAD vdd_1_inst ();
VSS PAD vss 1 inst ();
```

```
CORNER_PAD corner_2_inst ();

OUT_PAD data_out_12_inst (.PAD(data_out_pad[12]), .I(data_out[12]));

OUT_PAD data_out_13_inst (.PAD(data_out_pad[13]), .I(data_out[13]));

OUT_PAD data_out_14_inst (.PAD(data_out_pad[14]), .I(data_out[14]));

// ...

// Repeat

// ...

OUT_PAD data_out_146_inst (.PAD(data_out_pad[146]), .I(data_out[146]));

OUT_PAD data_out_147_inst (.PAD(data_out_pad[147]), .I(data_out[147]));

OUT_PAD data_out_148_inst (.PAD(data_out_pad[148]), .I(data_out[147]));

OUT_PAD data_out_149_inst (.PAD(data_out_pad[148]), .I(data_out[148]));

OUT_PAD data_out_149_inst (.PAD(data_out_pad[149]), .I(data_out[149]));

OUT_PAD data_out_150_inst (.PAD(data_out_pad[150]), .I(data_out[150]));

VDD PAD vdd 2 inst ();
```

VSS_PAD vss_2_inst ();

Step 1: Modify router LAXYZ.vnet (5/6)

OUT_PAD data_out_264_inst (.PAD(data_out_pad[264]), .I(data_out[264])); OUT_PAD data_out_265_inst (.PAD(data_out_pad[265]), .I(data_out[265]));

IN_PAD stop_in_0_inst (.PAD(stop_in_pad[0]), .O(stop_in[0])); IN_PAD stop_in_1_inst (.PAD(stop_in_pad[1]), .O(stop_in[1])); IN_PAD stop_in_2_inst (.PAD(stop_in_pad[2]), .O(stop_in[2])); IN_PAD stop_in_3_inst (.PAD(stop_in_pad[3]), .O(stop_in[3])); IN_PAD stop_in_4_inst (.PAD(stop_in_pad[4]), .O(stop_in[4])); IN_PAD stop_in_5_inst (.PAD(stop_in_pad[5]), .O(stop_in[5])); IN_PAD stop_in_6_inst (.PAD(stop_in_pad[6]), .O(stop_in[6]));

OUT_PAD stop_out_0_inst (.PAD(stop_out_pad[0]), .l(stop_out[0])); OUT_PAD stop_out_1_inst (.PAD(stop_out_pad[1]), .l(stop_out[1])); OUT_PAD stop_out_2_inst (.PAD(stop_out_pad[2]), .l(stop_out[2])); OUT_PAD stop_out_3_inst (.PAD(stop_out_pad[3]), .l(stop_out[3])); OUT_PAD stop_out_4_inst (.PAD(stop_out_pad[4]), .l(stop_out[3])); OUT_PAD stop_out_5_inst (.PAD(stop_out_pad[5]), .l(stop_out[4])); OUT_PAD stop_out_6_inst (.PAD(stop_out_pad[6]), .l(stop_out[6]));

Step 1: Modify router_LAXYZ.vnet (6/6)

- IN_PAD xaddr_0_inst (.PAD(xaddr_pad[0]), .O(xaddr[0]));
- IN_PAD xaddr_1_inst (.PAD(xaddr_pad[1]), .O(xaddr[1])); IN_PAD xaddr_2 inst (.PAD(xaddr_pad[2]), .O(xaddr[2]));
- IN PAD yaddr 0 inst (.PAD(yaddr pad[0]), .O(yaddr[0]));
- IN_PAD yaddr_1_inst (.PAD(yaddr_pad[1]), .O(yaddr[1]));
- IN_PAD yaddr_2_inst (.PAD(yaddr_pad[2]), .O(yaddr[2]));
- IN_PAD zaddr_0_inst (.PAD(zaddr_pad[0]), .O(zaddr[0]));
- IN_PAD zaddr_1_inst (.PAD(zaddr_pad[1]), .O(zaddr[1]));
- IN_PAD zaddr_2_inst (.PAD(zaddr_pad[2]), .O(zaddr[2]));
- IN_PAD clk_inst (.PAD(clk_pad), .O(clk));

```
IN_PAD reset_inst (.PAD(reset_pad), .O(reset));
```

```
VDD_PAD vdd_3_inst ();
```

```
VSS_PAD vss_3_inst ();
```

```
OR4_X1 U15 ( .A1(data_out[155]), .A2(data_out[154]), .A3(data_out[153]),
.A4(n3), .ZN(data_sent[4]) );
OR4_X1 U16 ( .A1(data_out[157]), .A2(data_out[156]), .A3(data_out[159]),
.A4(data_out[158]), .ZN(n3) );
```

```
// ...
```

Step 2: Make router_LAXYZ.io(1/5)

```
(globals
    version = 3
    io order = default
(iopad
    (topleft
        (inst name="corner 0 inst")
    )
    (top
        (inst name="data in 0 inst")
        (inst name="data in 1 inst")
        //... repeat ...
        (inst name="data in 137 inst")
        (inst name="data_in 138 inst")
        (inst name="vdd 0 inst")
        (inst name="vss 0 inst")
    (topright
        (inst name="corner 1 inst")
```

Step 2: Make router_LAXYZ.io(2/5)

(right

(inst name="data in 139 inst") (inst name="data in 140 inst") // ... repeat ... (inst name="data in 264 inst") (inst name="data in 265 inst") (inst name="data out 0 inst") (inst name="data out 1 inst") (inst name="data out 2 inst") (inst name="data out 3 inst") (inst name="data out 4 inst") (inst name="data out 5 inst") (inst name="data out 6 inst") (inst name="data out 7 inst") (inst name="data_out_8_inst") (inst name="data out 9 inst") (inst name="data out 10 inst") (inst name="data out 11 inst") (inst name="vdd 1 inst") (inst name="vss 1 inst")

Step 2: Make router_LAXYZ.io(3/5)

(bottomright

(inst name="corner_2_inst")

(bottom

(inst name="data_out_12_inst")

```
(inst name="data_out_13_inst")
```

// ... repeat ...

(inst name="data_out_149_inst")
(inst name="data_out_150_inst")
(inst name="vdd_2_inst")
(inst name="vss_2_inst")

)

(bottomleft

```
(inst name="corner_3_inst")
```

)

```
(left
```

```
(inst name="data_out_151_inst")
(inst name="data_out_152_inst")
// ... repeat ...
(inst name="data_out_264_inst")
(inst name="data_out_265_inst")
```

Step 2: Make router_LAXYZ.io(4/5)

(inst name="stop in 0 inst") (inst name="stop in 1 inst") (inst name="stop in 2 inst") (inst name="stop in 3 inst") (inst name="stop in 4 inst") (inst name="stop in 5 inst") (inst name="stop in 6 inst") (inst name="stop out 0 inst") (inst name="stop out 1 inst") (inst name="stop out 2 inst") (inst name="stop out 3 inst") (inst name="stop out 4 inst") (inst name="stop out 5 inst") (inst name="stop out 6 inst") (inst name="xaddr 0 inst") (inst name="xaddr 1 inst") (inst name="xaddr 2 inst")

Step 2: Make router_LAXYZ.io(5/5)

```
(inst name="yaddr_0_inst")
(inst name="yaddr_1_inst")
(inst name="yaddr_2_inst")
(inst name="zaddr_0_inst")
(inst name="zaddr_1_inst")
(inst name="zaddr_2_inst")
(inst name="clk_inst")
(inst name="reset_inst")
(inst name="vdd_3_inst")
(inst name="vss_3_inst")
```

Step 3: Make iopad_LAXYZ.tcl (1/9)

```
# Step 1: Setup (File --> Import Design)
```

```
#
```

#

setUIVar rda_Input ui_netlist ./input_files/router_LAXYZ.vnet setUIVar rda_Input ui_timingcon_file ./input_files/router_LAXYZ.sdc setUIVar rda_Input ui_topcell router_LAXYZ setUIVar rda_Input ui_leffile {~/lib/cells.lef ~/lib/iopad.lef} setUIVar rda_Input ui_timelib ~/lib/slow.lib setUIVar rda_Input ui_io_file ./input_files/router_LAXYZ.io setUIVar rda_Input ui_pwrnet VDD setUIVar rda_Input ui_gndnet VSS setUIVar rda_Input ui_cts_cell_list {CLKBUF_X1 CLKBUF_X2 CLKBUF_X3} commitConfig

Step 3: Make *iopad_LAXYZ.tcl(2/9)*

```
#
# Step 2: Floorplan (Floorplan --> Specify Floorplan)
#
floorPlan -s 150 150 15 15 15 15
saveDesign ./checkpoints/floor.enc
#
# Step 3: Power ring (Power --> Power Planning --> Add Ring)
#
addRing -nets {VSS VDD} -type core rings \
 -spacing top 2 -spacing bottom 2 -spacing right 2 -spacing left 2 \setminus
 -width top 4 -width bottom 4 -width right 4 -width left 4 \setminus
 -around core -jog distance 0.095 -threshold 0.095 \
 -layer top metal10 -layer bottom metal10 -layer right metal9 \
 -layer left metal9 \
 -stacked via top_layer metal10 -stacked_via_bottom_layer metal1
```

Step 3: Make iopad_LAXYZ.tcl(3/9)

Step 4: Power stripe (Power --> Power Planning --> Add Striple)

```
#
```

#

addStripe -nets {VSS VDD} -layer metal8 -width 4 -spacing 2 \ -block_ring_top_layer_limit metal9 -block_ring_bottom_layer_limit metal7 \ -padcore_ring_top_layer_limit metal9 -padcore_ring_bottom_layer_limit metal7 \ -stacked_via_top_layer metal10 -stacked_via_bottom_layer metal1 \ -set_to_set_distance 50 -xleft_offset 50 -merge_stripes_value 0.095 \ -max_same_layer_jog_length 1.6

Step 3: Make *iopad_LAXYZ.tcl(4/9)*

Step 5: Power route (Route --> Special Router)

#

#

```
globalNetConnect VDD -pin VDD -inst * -type pgpin
globalNetConnect VSS -pin VSS -inst * -type pgpin
```

sroute -nets {VSS VDD} -layerChangeRange {1 10} \

-connect { blockPin padPin padRing corePin floatingStripe } \

```
-blockPinTarget { nearestRingStripe nearestTarget } \
```

```
-padPinPortConnect { allPort oneGeom } \
```

```
-checkAlignedSecondaryPin 1 -blockPin useLef -allowJogging 1 \
```

```
-crossoverViaBottomLayer 1 -allowLayerChange 1 -targetViaTopLayer 10 \
```

```
-crossoverViaTopLayer 10 -targetViaBottomLayer 1
```

saveDesign ./checkpoints/power.enc

Step 3: Make *iopad_LAXYZ.tcl(5/9)*

```
#
# Step 6: Placement (Place --> Standard Cell)
#
placeDesign -prePlaceOpt
#
# Step 7: Optimization (preCTS) (Optimize --> Optimize Design)
#
optDesign -preCTS
#
# Step 8: Clock tree synthesis (CTS) (Clock --> Cynthesize Clock Tree)
#
addCTSCellList {CLKBUF X1 CLKBUF X2 CLKBUF X3}
clockDesign -genSpecOnly Clock.ctstch
clockDesign -specFile Clock.ctstch -outDir clock report -fixedInstBeforeCTS
saveDesign ./checkpoints/cts.enc
```

Step 3: Make *iopad_LAXYZ.tcl(6/9)*

```
#
# Step 9: Clock tree check (Clock --> Display --> Display Clock Tree)
#
#
# Step 9: Optimization (postCTS) (Optimize --> Optimize Design)
#
optDesign -postCTS
optDesign -postCTS -hold
```
Step 3: Make iopad_LAXYZ.tcl(7/9)

Step 10: Detailed route (Route --> Nano Route --> Route)

setNanoRouteMode -quiet -routeWithTimingDriven true setNanoRouteMode -quiet -routeTopRoutingLayer default setNanoRouteMode -quiet -routeBottomRoutingLayer default setNanoRouteMode -quiet -drouteEndIteration default setNanoRouteMode -quiet -routeWithTimingDriven true routeDesign -globalDetail

```
#
# Step 11: Optimization (postRoute) (Optimize --> Optimize Design)
#
optDesign -postRoute
optDesign -postRoute -hold
```

saveDesign ./checkpoints/route.enc

#

#

Step 3: Make *iopad_LAXYZ.tcl(8/9)*

```
#
# Step 12: Add fillers (Place --> Physical Cells --> Add Filler)
#
addFiller -prefix FILLER -cell FILLCELL X1 FILLCELL X2 FILLCELL X4 \
 FILLCELL X8 FILLCELL_X16 FILLCELL_X32
#
# Step 13: Verification (LVS) (Verify --> Verify Connectivity)
#
verifyConnectivity -type all -error 1000 -warning 50
#
# Step 14: Verification (DRC) (Verify --> Verify Geometry)
#
verifyGeometry
```

Step 3: Make iopad_LAXYZ.tcl(9/9)

Step 15: Data out (Timing --> Extract RC, Timing --> Write SDF,

File --> Save --> Netlist)

saveNetlist ./output/router_LAXYZ.vnet

isExtractRCModeSignoff

#

rcOut -spef ./output/router_LAXYZ.spef

delayCal -sdf ./output/router_LAXYZ.sdf -idealclock

saveDesign ./checkpoints/final.enc



Step 4: Script execution

Type velocity -init script/iopad_LAXYZ.tcl to start execute the script

%velocity script/par_LAXYZ.tcl

Type win to start SoC Encounter and visualize the final layout

%velocity script/par_LAXYZ.tcl



Final Layout

mize <u>Clock Route Timing V</u>erify Options Tools Flows <u>H</u>elp



Your final Chip layout will appear on the main window. Congratulations!



Final Layout

- If we zoom in, we can see the connection established between the pin, the pad and the signals wires connected to the inputports
- This figure shows the pads insertion for VDD, VSS, and the local input-port (data-in 0~11)





ACKNOWLEDGEMENT

<== Back to Contents

- THIS WORK IS SUPPORTED BY VLSI DESIGN AND EDUCATION CENTER (VDEC), THE UNIVERSITY OF TOKYO, JAPAN, IN COLLABORATION WITH SYNOPSYS, Inc. and CADENCE DESIGN SYSTEMS, Inc.
- THIS WORK IS SUPPORTED BY COMPETITIVE RESEARCH FUNDING, UNIVERSITY OF AIZU, JAPAN, Ref. P12-2013.