

# Post Verification After Place & Route

**Jaeha Kim, Sung-Joon Lee, and Eunseo Kim**  
**Mixed-Signal IC and System Group**  
**Seoul National University**  
**May 20. 2016**

# Generating Output

- ▶ After all process of P&R is finished, we can extract the result of P&R by typing :

```
$~/IDEC_CBDF/Place_Route/script> make output
```

- ▶ we'll get the following files in “results” folder
  - ▶ top.gds : Final GDS of our design
  - ▶ top.output.pg.lvs.v : Verilog model after P&R
  - ▶ top.output.spef.min(max) : SPEF file (parasitics information)
  - ▶ top.output.sdf : SDF file
  - ▶ ...

# Generating Output

► \$~/IDEC\_CBDF/Place\_Route/script> vi icc\_scripts/output.tcl

```
verify_drc -check_via_size ...
```

Running DRC & LVS in script

```
verify_lvs -max_error 100000...
```

```
write_verilog ... $DESIGN.output.pg.lvs.v
```

Write Verilog, SDF, SPEF, ...

```
#SDF
```

```
#SPEF
```

```
...
```

GDS stream out

```
write_stream -format gds -cells {top.CEL;#} $RESULTS_DIR/${DESIGN}.gds
```

↑ Type the latest version

# Post Top Model Generation

- Copy the following result files to the top model directory

```
$~/IDEC_CBDF/Place_Route/script> cd results  
$ cp top.output.sdf ~/IDEC_CBDF/xmodel_dp11/top  
$ cp top.output.pg.lvs.v ~/IDEC_CBDF/xmodel_dp11/top  
$ cp top.output.spef.max ~/IDEC_CBDF/xmodel_dp11/top
```

- Run `post_netlist.py` to generate post top model (top\_post.sv)

```
$ cd ~/IDEC_CBDF/xmodel_dp11/top  
$~/IDEC_CBDF/xmodel_dp11/top> python post_netlist.py top.f  
top.output.pg.lvs.v top.output.spef.max
```

# Final Top Simulation

- ▶ The final simulation includes
  - ▶ Top simulation (DPLL + PAD)
  - ▶ Digital loop filter timing information
  - ▶ Delay elements induced by wiring resistance & capacitance in P&R (interconnection between blocks)
- ▶ This simulation will estimate the real chip performance
- ▶ To simulate it, all you need is ...
  - ▶ Post top model after P&R : **top\_post.sv**
  - ▶ SDF file of the top model : **top.output.sdf**
  - ▶ Digital and pad library : **Library/verilog/, xmodel\_dpll/viola/**

# Exercise: Final Chip Simulation

- ▶ Testbench: 'tb\_locking'
- ▶ First, check if there's top\_post.sv is located at the correct directory

```
$ cd ~/IDEC_CBDF/xmodel_dpll/top  
$ ls
```

- ▶ Change the directory to 'tb\_locking'

```
$~/IDEC_CBDF/xmodel_dpll/top> cd tb/tb_locking
```

# Exercise: Final Chip Simulation (2)

- ▶ Open the tb\_locking.sv

```
$~/IDEC_CBDF/xmodel_dp11/top/tb/tb_locking> vi tb_locking.sv
```

- ▶ **Important**

- ▶ You need to fix sdf path to the right “absolute” path of the target .sdf file (ex. top.output.sdf)

```
initial begin
`ifdef SDF
    $sdf_annotate("/afs/mics.snu.ac.kr/user/eunseo/IDEC_CBDF/xmodel_dp11/top
/top.output.sdf",DUT);
    $vcdplusfile ("tb_locking.sdf.vpd");
`else
    $vcdplusfile("tb_locking.vpd");
`endif
    $vcdpluson;
end
```

# Run the Simulation

- ▶ Run the simulation by typing 'make <mode>'

```
$ make tb_locking_sdf
```

- ▶ (If you type 'make tb\_locking', the simulation result will be the same with what you did in the GLISTER environment)
- ▶ The simulation is now running



# SDF Annotation Checking

- ▶ You need to check the following at your terminal (while in simulating)

```

31 2016
Doing SDF annotation ..... Done

  | |/_/ |/_/ |/_/ |/_/ |/_/ |/_/ |/_/ |/_/ |/_/ (TM)
  _| |/_/ |/_/ |/_/ |/_/ |/_/ |/_/ |/_/ |/_/ |/_/
/_/|/_/|/_/ |/_/|/_/|/_/|/_/|/_/|/_/|/_/|/_/ ANALOG/MIXED-SIGNAL SIMULATOR

XMODEL Release 2016.0517 (x86_64)
Copyright (c) 2012-2015 Scientific Analog, Inc.
All rights reserved and patents pending.

#LMX: LM-X feature 'XMODEL' has been successfully checked out.
#LMX: Version: 2.1 (academic)
#LMX: Expiration: 2016-12-31 23:59
#LMX: License type: network
#LMX: License server: arctic

NOTICE: this software is licensed for research and educational
purposes only and not permitted for commercial, income-producing
activities.

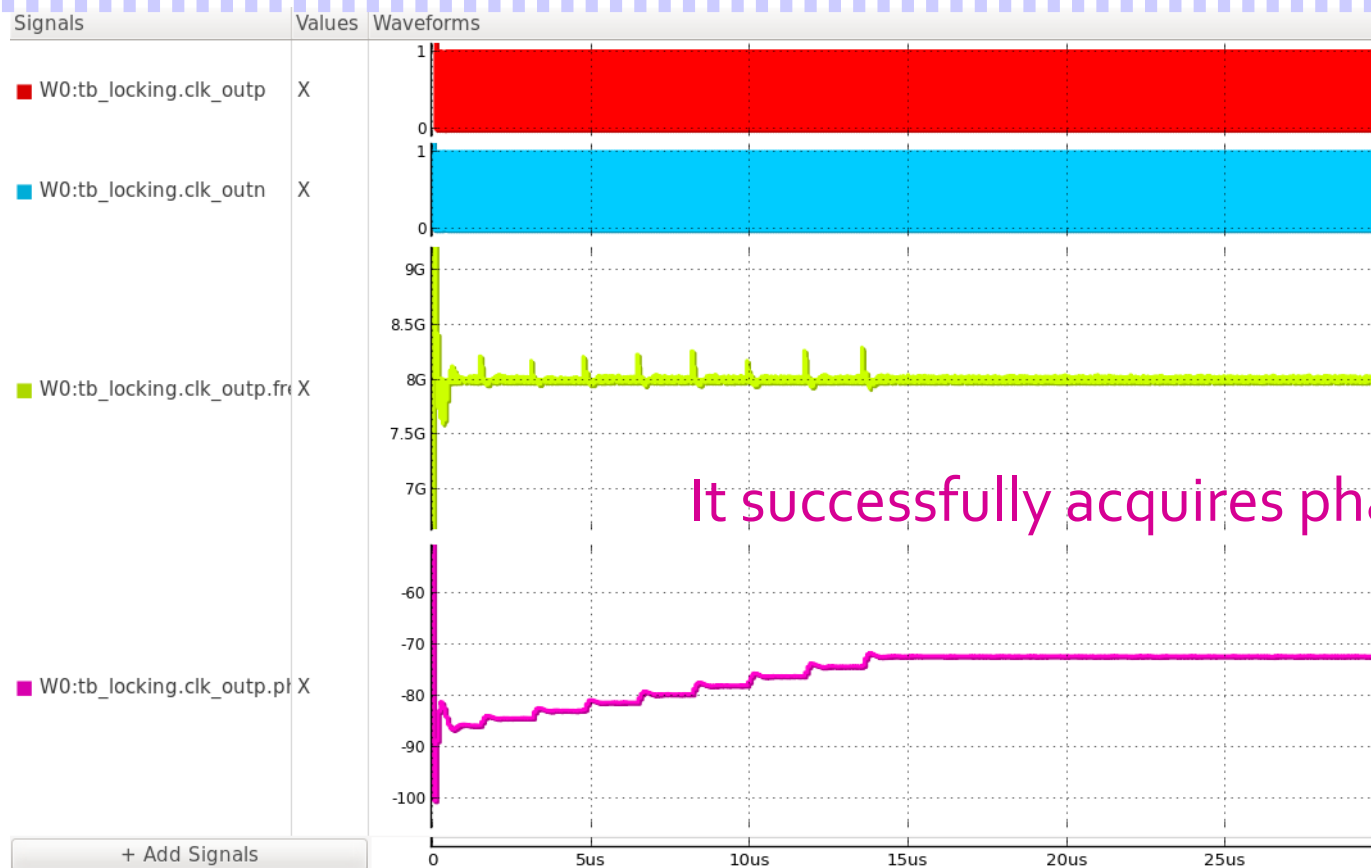
VCD+ Writer H-2013.06_Full64 Copyright (c) 1991-2013 by Synopsys Inc.

```

# Result - Locking

## ► Open XWAVE waveform viewer

```
$ xwave xmodel.jez
```



It successfully acquires phase-lock

# Result – Jitter Histogram

- ▶ You can also plot jitter histogram in this testbench
  - ▶ Estimated output clock RMS jitter = 420 fs
  - ▶ (20 ~ 30 us)

