Genus Tutorial

NORTHWESTERN
UNIVERSITY

McCormick

Northwestern Engineering

Genus Tutorial

Before going to next steps, please note that those lines that start with '#' are explanation, lines that follow with '$' are commands and you need to copy and then paste in your terminal and press enter.

1) # Type following command to source the cadence environment.

   $ source /vol/ece303/genus_tutorial/cadence.env

```
[qcb2982@ras ~]$ source /vol/ece303/genus_tutorial/cadence.env
[qcb2982@ras ~]$ 
```

2) # Create working directory, for example "Lab2". Then go into the directory "Lab2" and copy files to this folder. 'mkdir' means make directory; 'cd' means change directory; 'cp' means copy.

   $ mkdir Lab2

   $ cd ./Lab2

   $ cp /vol/ece303/genus_tutorial/alu_conv.v .

   $ cp /vol/ece303/genus_tutorial/alu_conv.sdc .

   $ mkdir Synthesis

   Lab2 folder should contain: alu_conv.sdc, alu_conv.v, Synthesis. You could type "ls" to see files in the directory.

3) # Go to **Synthesis** folder and then type "genus" and press enter to run the cadence tool.

   $ cd Synthesis

   $ genus

```
[qcb2982@ras Synthesis]$ genus
TMPDIR is being set to /tmp/genus_temp_19276_ras.ece.northwestern.edu_qcb2982_pvUhwk
Cadence Genus(TM) Synthesis Solution.
Copyright 2018 Cadence Design Systems, Inc. All rights reserved worldwide.
Cadence and the Cadence logo are registered trademarks and Genus is a trademark
of Cadence Design Systems, Inc. in the United States and other countries.

Version: 18.14-s037_1, built Wed Mar 27 10:19:21 PDT 2019
Options:
Date:    Wed Sep 18 16:31:15 2019
Host:    ras.ece.northwestern.edu (x86_64 w/Linux 2.6.32-754.22.1.el6.x86_64) (4cores*8cpus*1physical cpu*In
tel(R) Xeon(R) CPU E5-1620 0 @ 3.60GHz 10240KB) (16252572KB)
OS:      Red Hat Enterprise Linux Server release 6.10 (Santiago)

Checking out license: Genus_Synthesis

Loading tool scripts...

Finished loading tool scripts (11 seconds elapsed).

@genus:root: 1> 
```

   Important: Everything will be command based. There is no GUI interface.

4) # **read** RTL file, '../' refers to files in the upper level folder

   $ read_hdl ../alu_conv.v

```
@genus:root: 1> read_hdl ../alu_conv.v
@genus:root: 2> 
```

5) # **set** lib and lef files. This provides information of gates.

   $ set_db library /vol/ece303/genus_tutorial/NangateOpenCellLibrary_typical.lib

   $ set_db lef_library /vol/ece303/genus_tutorial/NangateOpenCellLibrary.lef

   Important: It's fine to have Warning, but ERROR needs to be fixed.

```
@genus:root: 2> set_db library /vol/ece303/genus_tutorial/NangateOpenCellLibrary_typical.lib
```

```
@genus:root: 3> set_db lef_library /vol/ece303/genus_tutorial/NangateOpenCellLibrary.lef
```

6) #**Elaborating** design

$ elaborate

$ current_design alu_conv

```
@genus:root: 4> elaborate
```

```
@genus:root: 5> current_design alu_conv
design:alu_conv
```

7) #**Rea**d sdc file, which is for timing constraints. It defines max delay, load, max capacitance and max transition of the circuit. '../' refers to files in the upper level folder

$ read_sdc ../alu_conv.sdc

Important: read_sdc should have 0 failed

```
@genus:design:alu_conv 6> read_sdc alu_conv.sdc
Statistics for commands executed by read_sdc:
 "all_inputs"              - successful      2 , failed      0 (runtime  0.00)
 "all_outputs"             - successful      2 , failed      0 (runtime  0.00)
 "current_design"          - successful      1 , failed      0 (runtime  0.00)
 "set_load"                - successful      1 , failed      0 (runtime  0.00)
 "set_max_capacitance"     - successful      1 , failed      0 (runtime  0.00)
 "set_max_delay"           - successful      1 , failed      0 (runtime  0.00)
 "set_max_transition"      - successful      1 , failed      0 (runtime  0.00)
Total runtime 0.0
@genus:design:alu_conv 7>
```

8) # The setting below are all default. Syn_generic is for synthesis into generic gates with some RTL optimization. Syn_map is for mapping the design to cells from provided library with some logic optimization. Syn_opt is for gate level optimization. Then starts to Synthesize.

$ syn_generic

$ syn_map

$ syn_opt

```
@genus:design:alu_conv 7> syn_generic
```

```
@genus:design:alu_conv 8> syn_map
```

```
@genus:design:alu_conv 9> syn_opt
```

9) # **Repor**t timing, and it will return longest logic path, and the timing slack of the design.

$ report_timing > timing.rpt

```
@genus:design:alu_conv 10> report_timing > timing.rpt
```

```
Path 1: MET (377 ps) Path Delay Check
     Startpoint: (F) a_sel
       Endpoint: (F) out[7]

                       Capture    Launch
      Path Delay:+       1000          -
      Drv Adjust:+          0          0
         Arrival:=       1000

   Required Time:=       1000
       Data Path:-        623
           Slack:=        377
```

After timing report, you can open timing.rpt file using a text editor, e.g. vim or emacs, to see the report above. You can open another terminal to do this if you do not want to quit the current Genus session.

NOTE THAT THE ACTUAL DELAY AND AREA VALUES YOU OBSERVE WHEN YOU RUN GENUS MAY DIFFER FROM THOSE IN THESE SCREENSHOTS AS THE LIBRARIES AND SOFTWARE GET UPDATES OVER TIME.

10) # **Report** area, and it will return the design name, design area, and gate numbers.

$ report_area > area.rpt

```
@genus:design:alu_conv 11> report_area > area.rpt
```

```
Instance Module  Cell Count  Cell Area  Net Area   Total Area
--------------------------------------------------------------
alu_conv                220    225.834   322.806      548.640
```

Note: After area report, you can open area.rpt file using a text editor to see report above.

11) # **Write** gate level netlist file. This is finally generated gate level netlist from synthesis.

$ write_hdl > alu_conv_syn.v

```
@genus:design:alu_conv 12> write_hdl > alu_conv_syn.v
```

12) # **Quit** from tool

$ quit

```
@genus:design:alu_conv 13> quit
```

Note in this design example, the design is fully combinational without sequential circuits and clocks. For a design with clock, the sdc file needs to be modified to define clock period. Examples can be found in the commented-out lines in the sdc file, i.e. alu_conv.sdc.