ESE 566A Modern System-on-Chip Design, Spring 2017
Tutorial for Synopsys Design Compiler

Electrical and System Engineering, Washington University

Revision: 01-25-2016

1. Introduction

The process that Design Compiler does is RTL synthesis. This means, converting a gate level logic Verilog file to transistor level Verilog with the help of Technology library provided by the foundry.

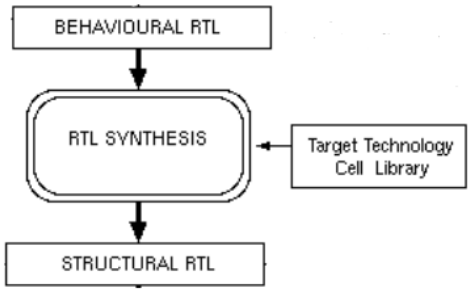

Figure 1.1 Workflow of DC

We use Synopsys Design Compiler (DC) to synthesize Verilog RTL models into a gate-level netlist where all of the gates are from the standard cell library. So Synopsys DC will synthesize the Verilog + operator into a specific arithmetic block at the gate-level. Based on various constraints it may synthesize a ripple-carry adder, a carry-look-ahead adder, or even more advanced parallel-prefix adders.

2. Login to the Linux Lab server

Detailed explanation is in *ese566-linux-tutorial.pdf*

3: Getting started with Verilog

Creating a new folder (better if you have all the files for a project in a specific folder). Then, enter into this new folder and start writing your Verilog script in a new file (.v file). Example code for modeling a counter is here. In addition to model code, Test Bench script has to be given in order to verify the functionality of your model (.v file). Example code of test bench for counter is here.

4: Writing constraints file - ".tcl" file

Example tcl file for counter above is here.
Based on the example tcl file, Lines where modifications are required specific to model:

- Set Path to Verilog files
- Top module of the design have to be specified (To specify the Hierarchy)
- Specifying the clock port used in the model
- Clock frequency at which the model is operated

Figure 4.1 Edit tcl file using gedit

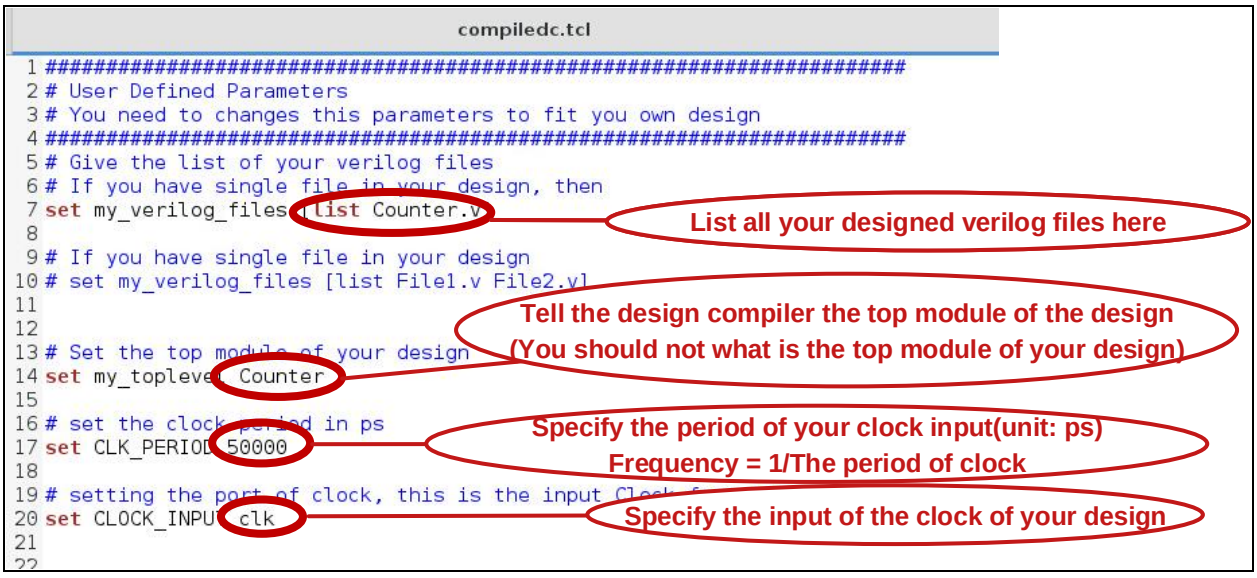

Fig. 4.2 Lines that modifications are required

Note: Please go through the provided *.tcl* file to check other options by yourself.

5. Compile the code

In the terminal, change the directory to where your model (*Counter.v*) and tcl(*compiledc.tcl*) files are present by using this command:

% cd ~/ese566A/DCTutorial/

Compile the files by typing in the terminal:

% dc_shell-t -f <file>.tcl

In the above example, it should be:

% dc_shell-t -f compiledc.tcl

After run this command, there might be some warning but no error presented in the terminal. Otherwise you need to check your code or *tcl* file and correct them according to the related messages.

Note: The oldest messages might be lost due to too many lines printed on the terminal. In this situation you could redirect the output to a file:

% dc_shell-t -f compiledc.tcl > compiledc.log

Then you can open *compiledc.log* using *gedit* and search in it. The “>” here is the sign for *Output Redirection* in Linux shell.

6. Result

After successfully compile the code, you will find the report files(name of the files are defined in *tcl* file above) in your project folder.

Note: *<my_toplevel>* following is defined in *tcl* file. In the above example *<my_toplevel>* is *Counter*.

6.1 Timing Analysis reports

- *<my_toplevel>_min_timing.repC*
- *<my_toplevel>_max_timing.repC*
- *<my_toplevel>_out_min_timing.repC*

Make sure the timing report requirements are MET. You can observe which module in the design is giving the maximum delay and optimize accordingly.

```
****************************************
Report : timing
         -path full
         -delay min
         -nworst 3
         -greater_path 0.00
         -max_paths 20
Design : Counter
Version: J-2014.09-SP5
Date   : Sun Oct  2 17:39:06 2016
****************************************

Operating Conditions: nom_pvt   Library: vtvt_tsmc180
Wire Load Model Mode: top

  Startpoint: c_reg[0] (rising edge-triggered flip-flop clocked by clk)
  Endpoint: c_reg[0] (rising edge-triggered flip-flop clocked by clk)
  Path Group: clk
  Path Type: min

  Point                                    Incr       Path
  -----------------------------------------------------------
  clock clk (rise edge)                    0.00       0.00
  clock network delay (ideal)              0.00       0.00
  c_reg[0]/ck (dp_1)                       0.00       0.00 r
  c_reg[0]/q (dp_1)                      326.98     326.98 r
  U20/op (nor2_1)                        100.13     427.12 f
  c_reg[0]/ip (dp_1)                       0.00     427.12 f
  data arrival time                                 427.12

  clock clk (rise edge)                    0.00       0.00
  clock network delay (ideal)              0.00       0.00
  c_reg[0]/ck (dp_1)                       0.00       0.00 r
  library hold time                        0.00       0.00
  data required time                                  0.00
  -----------------------------------------------------------
  data required time                                  0.00
  data arrival time                                -427.12
  -----------------------------------------------------------
  slack (MET)                                       427.12
```

Figure 6.1 Partial of timing report(Counter_min_timing.repC)

6.2 Power Analysis reports

- *<my_toplevel>_power.repC*

Design Compiler gives the detailed information about the static and dynamic power.

```
Global Operating Voltage = 1.8
Power-specific unit information :
    Voltage Units = 1V
    Capacitance Units = 1.000000ff
    Time Units = 1ps
    Dynamic Power Units = 1mW    (derived from V,C,T units)
    Leakage Power Units = 1mW


  Cell Internal Power  =  11.4299 uW   (77%)
  Net Switching Power  =   3.4778 uW   (23%)
                         ---------
Total Dynamic Power    =  14.9077 uW  (100%)

Cell Leakage Power     =  11.2191 nW
```

Figure 6.2 Partial of power report(Counter_power.repC)

6.3. Area Analysis reports

- *<my_toplevel>_area.repC*

Area report file generated using design compiler contains detail information about the size of each cell used for this model. Units for Virginia tech library are micro meters.

```
****************************************
Report : area
Design : Counter
Version: J-2014.09-SP5
Date   : Sun Oct  2 17:39:06 2016
****************************************

Library(s) Used:

    vtvt_tsmc180 (File: /project/linuxlab/cadence/vendors/VTVT/vtvt_tsmc180/Synopsys_Libraries/libs/vtvt_tsmc180.db)

Number of ports:                          7
Number of nets:                          27
Number of cells:                         25
Number of combinational cells:           20
Number of sequential cells:               5
Number of macros/black boxes:             0
Number of buf/inv:                        4
Number of references:                     8

Combinational area:               836.746197
Buf/Inv area:                     111.099602
Noncombinational area:            872.612991
Macro/Black Box area:               0.000000
Net Interconnect area:      undefined  (No wire load specified)

Total cell area:                 1709.359188
Total area:                 undefined
1
```

Figure 6.3 Partial of area report(Counter_area.repC)

6.4 Design Analysis reports

- *<my_toplevel>_design.repC*

Important information found in this report is the operating conditions. VT library only support normal conditions for operation. But in technology libraries provided by the vendor there are some extreme conditions (WCCO – worst case conditions) available for getting the maximum values to verify the performance of the model.

```
27  Operating Conditions:
28
29
30      Operating Condition Name : nom_pvt
31      Library : vtvt_tsmc180
32      Process :   1.00
33      Temperature :  25.00
34      Voltage :   1.80
35      Interconnect Model : balanced_tree
36
```

Figure 6.4 Partial of design report(Counter_area.repC)

7. Reference

[1] https://cornell-ece5745.github.io/ece5745-tut5-asic-tools/
[2] https://cornell-ece5745.github.io/ece5745-tut6-asic-flow/
[3] http://classes.engineering.wustl.edu/ese566/Tutorial/ese566-linux-tutorial.pdf
[4] http://classes.engineering.wustl.edu/ese566/Tutorial/ese566-vcs-tutorial.pdf
[5] http://www.csl.cornell.edu/courses/ece5745/handouts.html