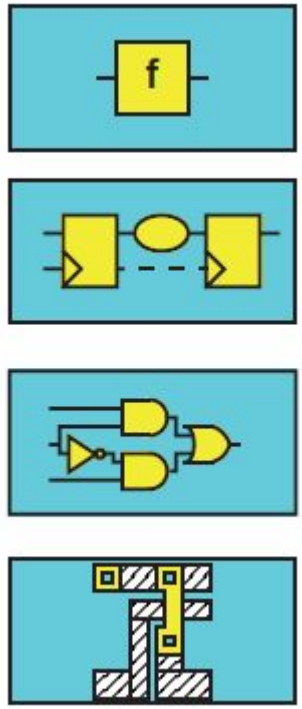
Outline

- Introduction
- Verilog for RTL
  - Module Description and Declaration
  - Data Type and Operators
  - Combinational Behavior
  - Sequential Behavior
  - Finite State Machine
  - Advanced Topics
- Design Example
- RTL Simulation Tool
- **Synthesis Tool** Part 3

Verilog-Supported Levels of Abstraction

- **Behavior Level**
  - Modeling the circuit's behavior in high-level.
- **Register Transfer Level (RTL)**
  - Describes the flow of data between registers and how a design process the data.
- **Gate Level** Using Design Vision
  - Describes the logic gates and the interconnections.
- **Transistor Level**
  - Describes the transistors and the interconnections.



Setup Environment

- **Source file provided by lab 231**

```
%  source synthesis.cshrc (助教提供)
```

- **Launch “design vision” for synthesis**

```
%  dv &
```

```
%  dc_shell
```

  - Remember execute the DV in the folder that has **.synopsys_dc.setup**

Setup Environment

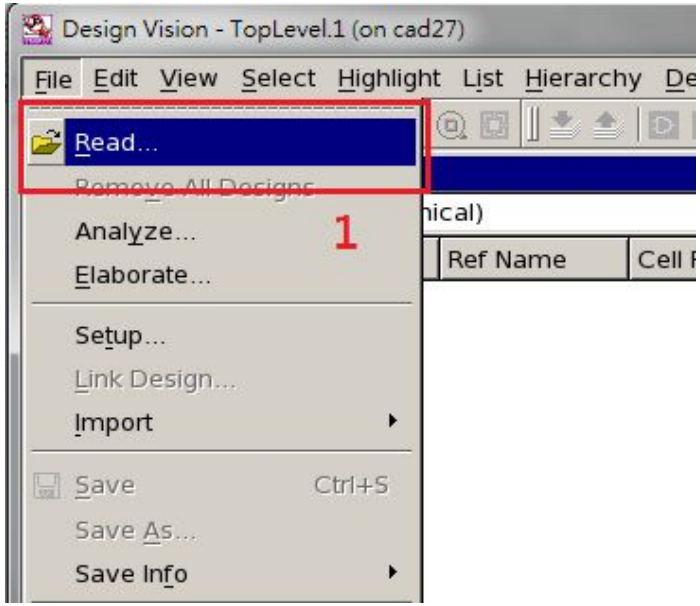
- **Please check the .synopsys_dc.setup file. It contains the path of cell library and related library.**
- **This file is hidden. Be careful that there is a “.” in front of the name.**

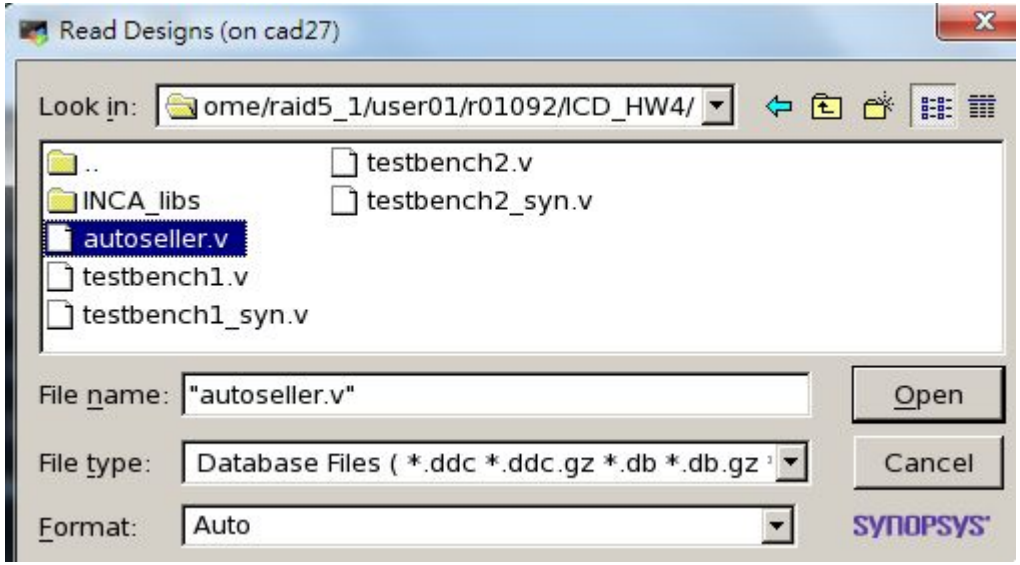
Design vision (Step by Step)

1. Read Design
2. Set Constraints
3. Check and Compile Design
4. Write reports
5. Write Essential Simulation Files

1. Read Design

- **Read the input Verilog file**
  - File->Read
  - Choose the design
  - Open





1. Read Design

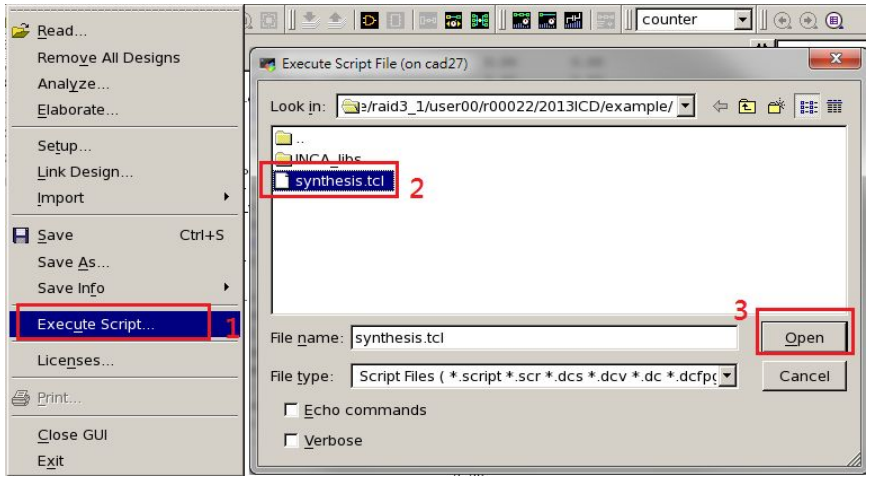
- **You should have a good coding style to avoid latch.**
- **Fully Assignment !!**

```
Inferred memory devices in process
        in routine counter line 61 in file
                '/home/raid3_1/user00/r00022/2013ICD/example/counter.v'.
===============================================================================
|    Register Name     |    Type    | Width | Bus | MB | AR | AS | SR | SS | ST |
===============================================================================
|      count_reg       | Flip-flop  |   2   |  Y  | N  | N  | Y  | N  | N  | N  |
|      count_reg       | Flip-flop  |   2   |  Y  | N  | Y  | N  | N  | N  | N  |
|       num_reg        | Flip-flop  |   2   |  Y  | N  | N  | Y  | N  | N  | N  |
|       num_reg        | Flip-flop  |   2   |  Y  | N  | Y  | N  | N  | N  | N  |
|      state_reg       | Flip-flop  |   2   |  Y  | N  | Y  | N  | N  | N  | N  |
===============================================================================
Presto compilation completed successfully.
Current design is now '/home/raid3_1/user00/r00022/2013ICD/example/counter.db:counter'
Loaded 1 design.
Current design is 'counter'.
design_vision>
Current design is 'counter'.
```

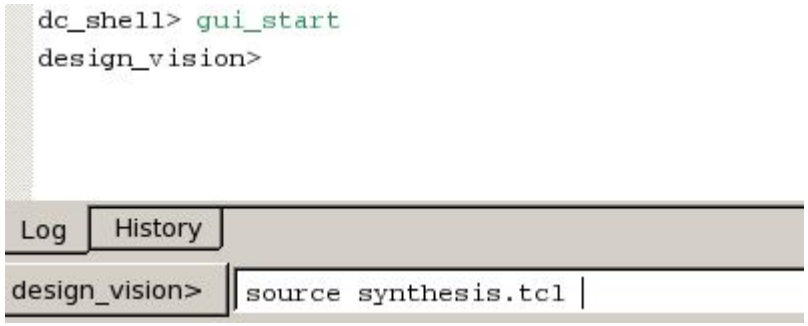
No Latch

2. Set Constraints

- **Execute the script (sythesis.tcl) written by TA**
  - Method 1



  - Method 2

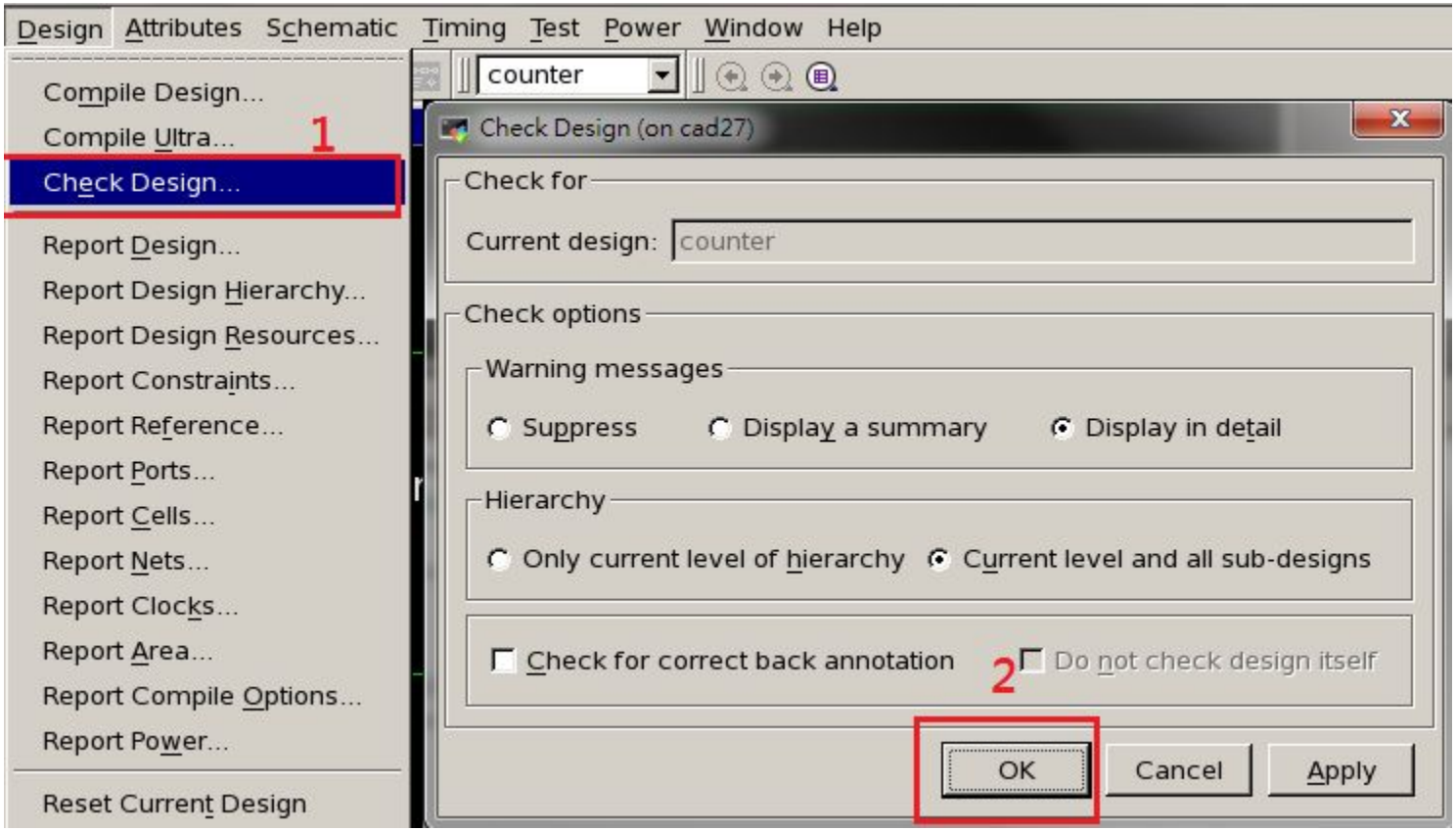


2. Set Constraints

```
#Setting Constraints
set cycle  10 ;#clock period defined by designer

create_clock -period $cycle          [get_ports  clk]
set_dont_touch_network               [get_clocks clk]
set_fix_hold                         [get_clocks clk]
set_clock_uncertainty        0.1     [get_clocks clk]
set_clock_latency   -source  0       [get_clocks clk]
set_clock_latency            1       [get_clocks clk]
set_input_transition         0.5     [all_inputs]
set_clock_transition         0.5     [all_clocks]

set_operating_conditions -min fast  -max slow

set_input_delay   -max 1    -clock clk   [all_inputs]
set_input_delay   -min 0.2  -clock clk   [all_inputs]
set_output_delay  -max 1    -clock clk   [all_outputs]
set_output_delay  -min 0.1  -clock clk   [all_outputs]
set_wire_load_model -name tsmc13_wl10 -library slow
```

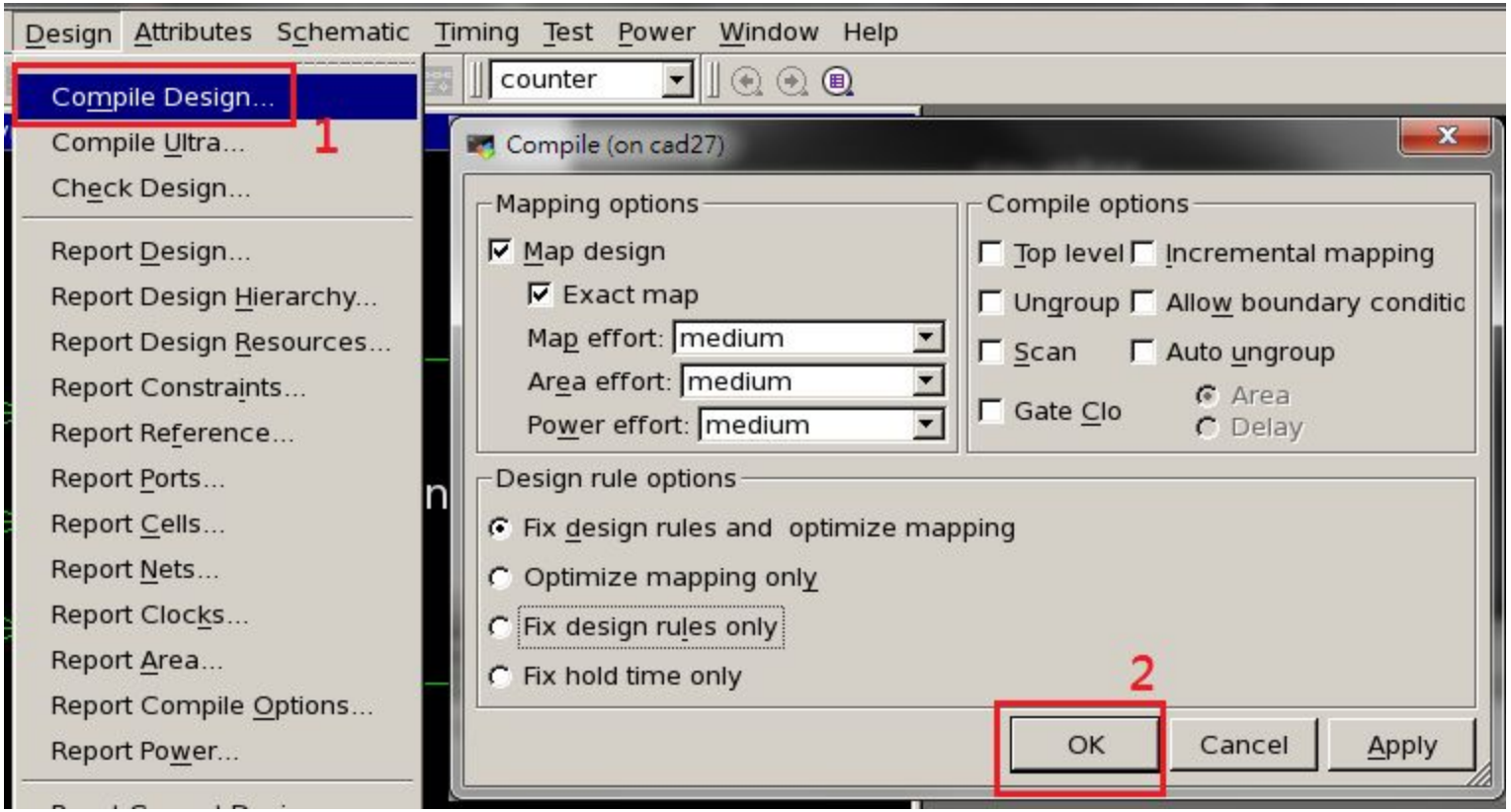
3. Check and Compile Design

- **Check Design**



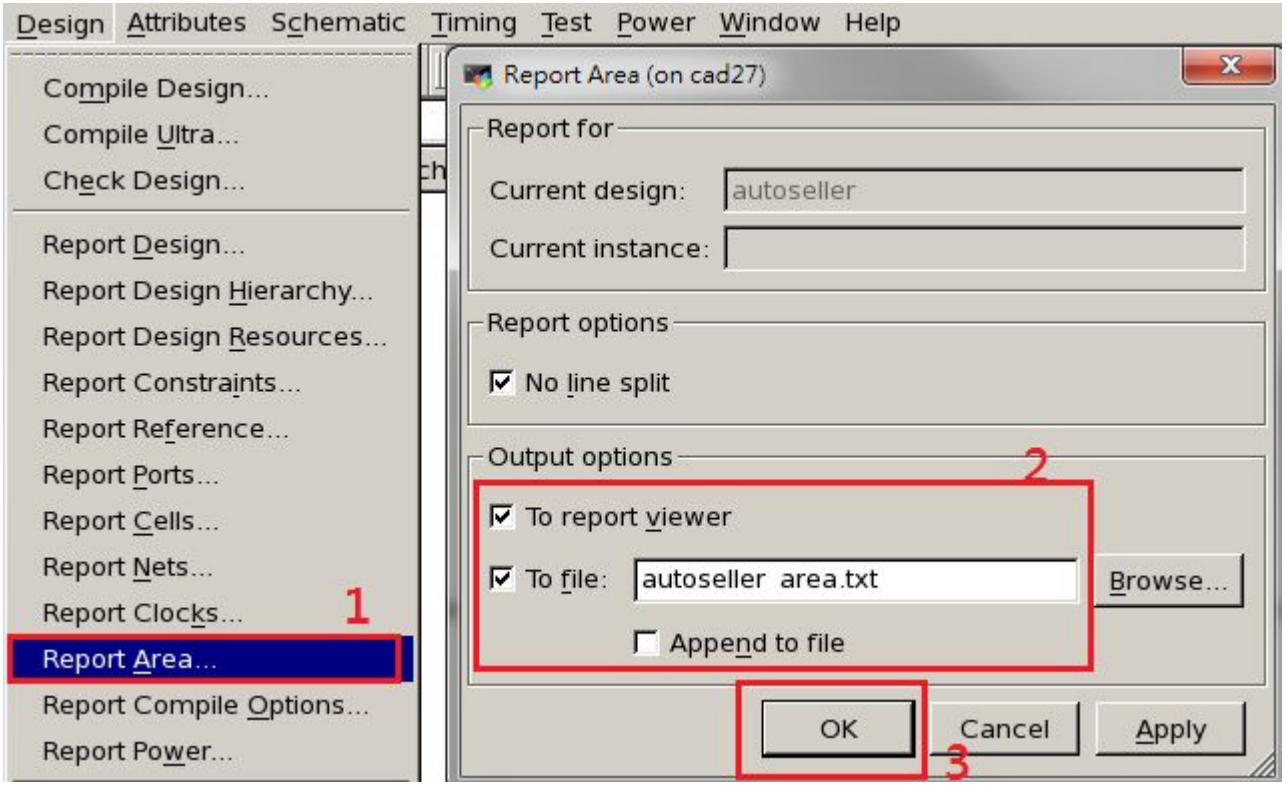
3. Check and Compile Design

- **Compile Design**
  - You can change the Map/Area/Power effort according to the spec.



4. Write reports

- **Report Area: Design/Report Area…**



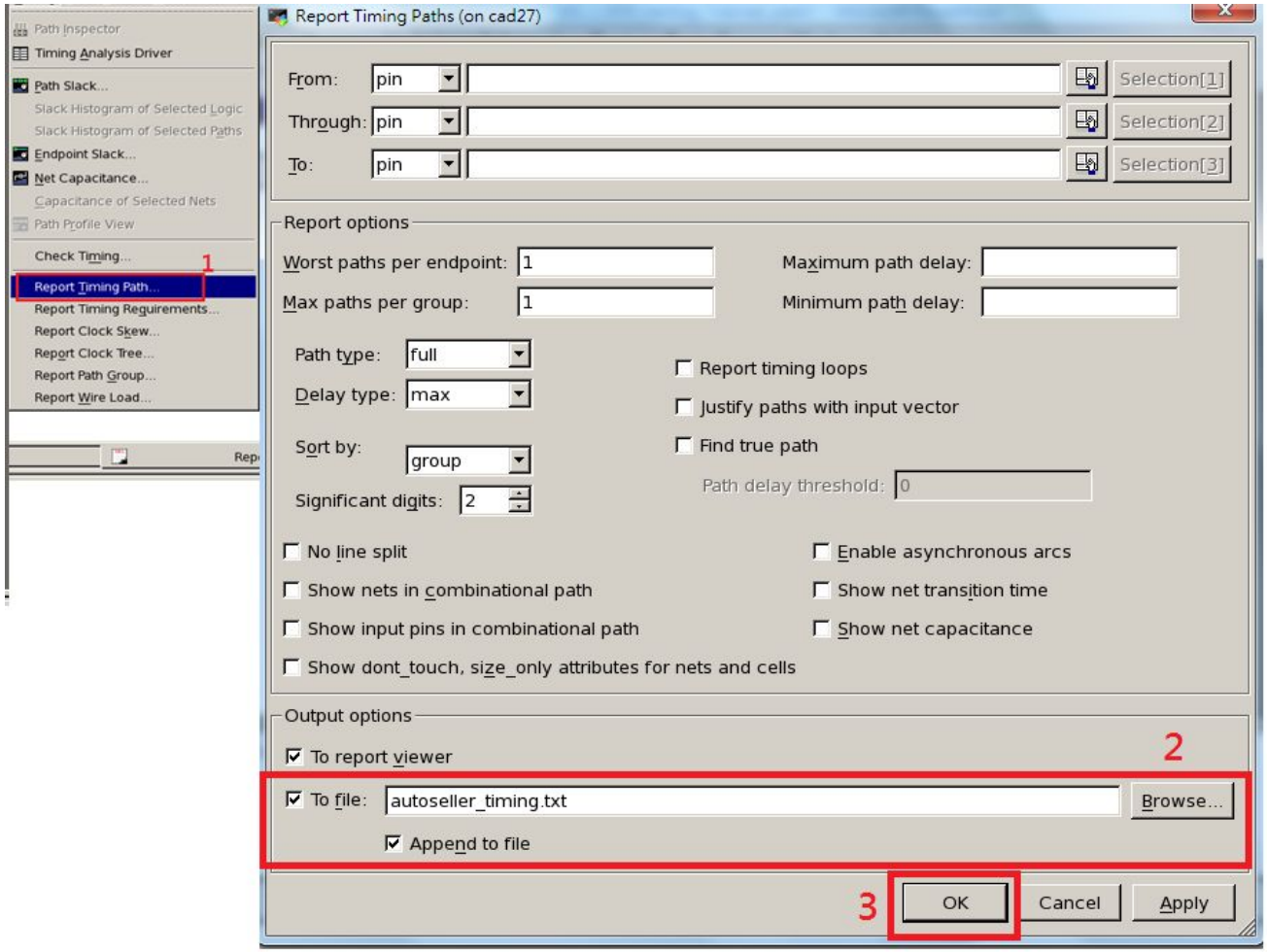
4. Write reports

- **Combinational area**
- **Noncombinational area**
- **Net Interconnect area**
  - depends on the wireload model you used and
  - The Net Interconnect area doesn't contribute wires are mostly lie on the standard cells.

```
****************************************
Report : area
Design : autoseller
Version: G-2012.06
Date   : Mon May 12 02:29:52 2014
****************************************

Library(s) Used:

    typical (File:

Number of ports:                        21
Number of nets:                        132
Number of cells:                        97
Number of combinational cells:          77
Number of sequential cells:             20
Number of macros:                        0
Number of buf/inv:                      12
Number of references:                   24

Combinational area:         831.726007
Noncombinational area:      641.617180
Net Interconnect area:    13113.398590

Total cell area:           1473.343187
Total area:               14586.741777
```

4. Write reports

- **Report Timing: Timing/Report Timing path…**



4. Write reports

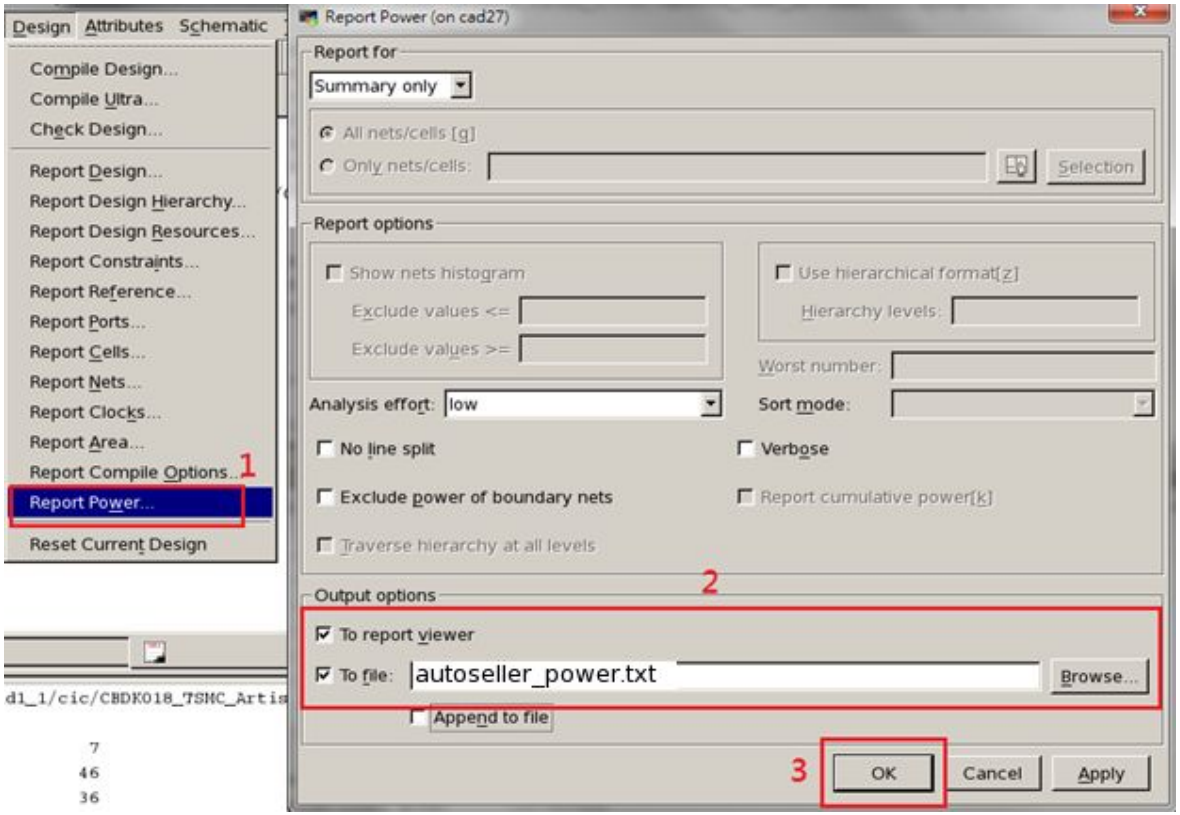
- **Timing Report**
  - Positive timing slack means the critical
  - Positive timing slack doesn't guarantee

Point	Incr	Path
clock clk (rise edge)	0.00	0.00
clock network delay (ideal)	1.00	1.00
money_reg[2]/CK (DFFRX1)	0.00	1.00 r
money_reg[2]/Q (DFFRX1)	0.71	1.71 r
U125/Y (NAND3X1)	0.24	1.95 f
U121/Y (OAI211X1)	0.43	2.38 r
U64/Y (CLKINVX1)	0.32	2.70 f
sub_109_aco/U2_1/CO (ADDFXL)	0.53	3.23 f
sub_109_aco/U2_2/CO (ADDFXL)	0.31	3.54 f
sub_109_aco/U2_3/CO (ADDFXL)	0.37	3.92 f
U73/Y (OR2X1)	0.22	4.14 f
U70/Y (XNOR2X1)	0.13	4.27 f
U96/Y (AOI221XL)	0.36	4.63 r
U95/Y (OAI22XL)	0.17	4.80 f
change_o_reg[5]/D (DFFRXL)	0.00	4.80 f
data arrival time		4.80
clock clk (rise edge)	10.00	10.00
clock network delay (ideal)	1.00	11.00
clock uncertainty	-0.10	10.90
change_o_reg[5]/CK (DFFRXL)	0.00	10.90 r
library setup time	-0.15	10.75
data required time		10.75
data required time		10.75
data arrival time		-4.80
slack (MET)		5.95

Positive Timing Slack!

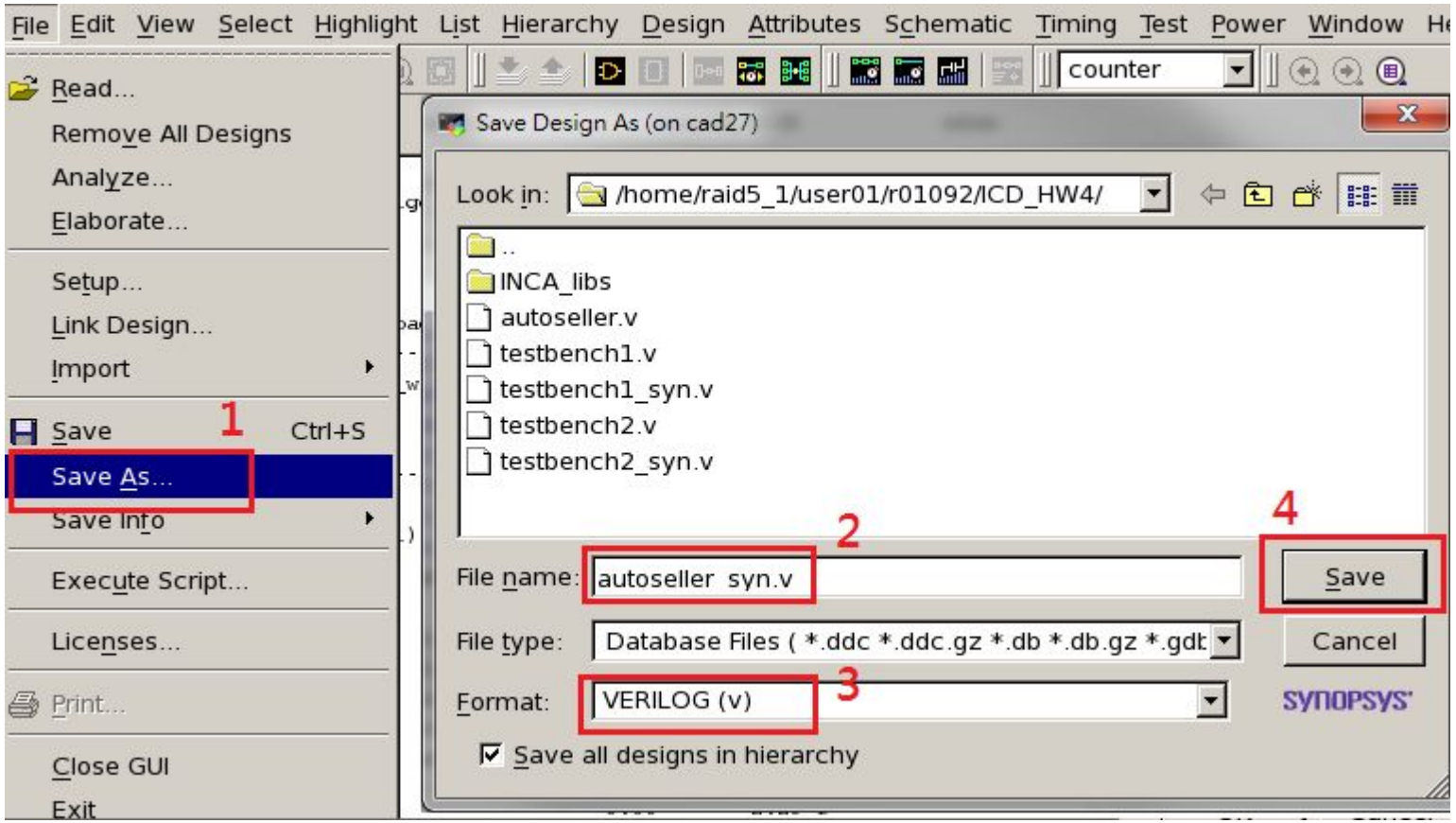
4. Write reports

- **Report Power: Design/Report Power…**
  - The power report in Design Vision is a rough estimation, the real power consumption depends on the input and output patterns.



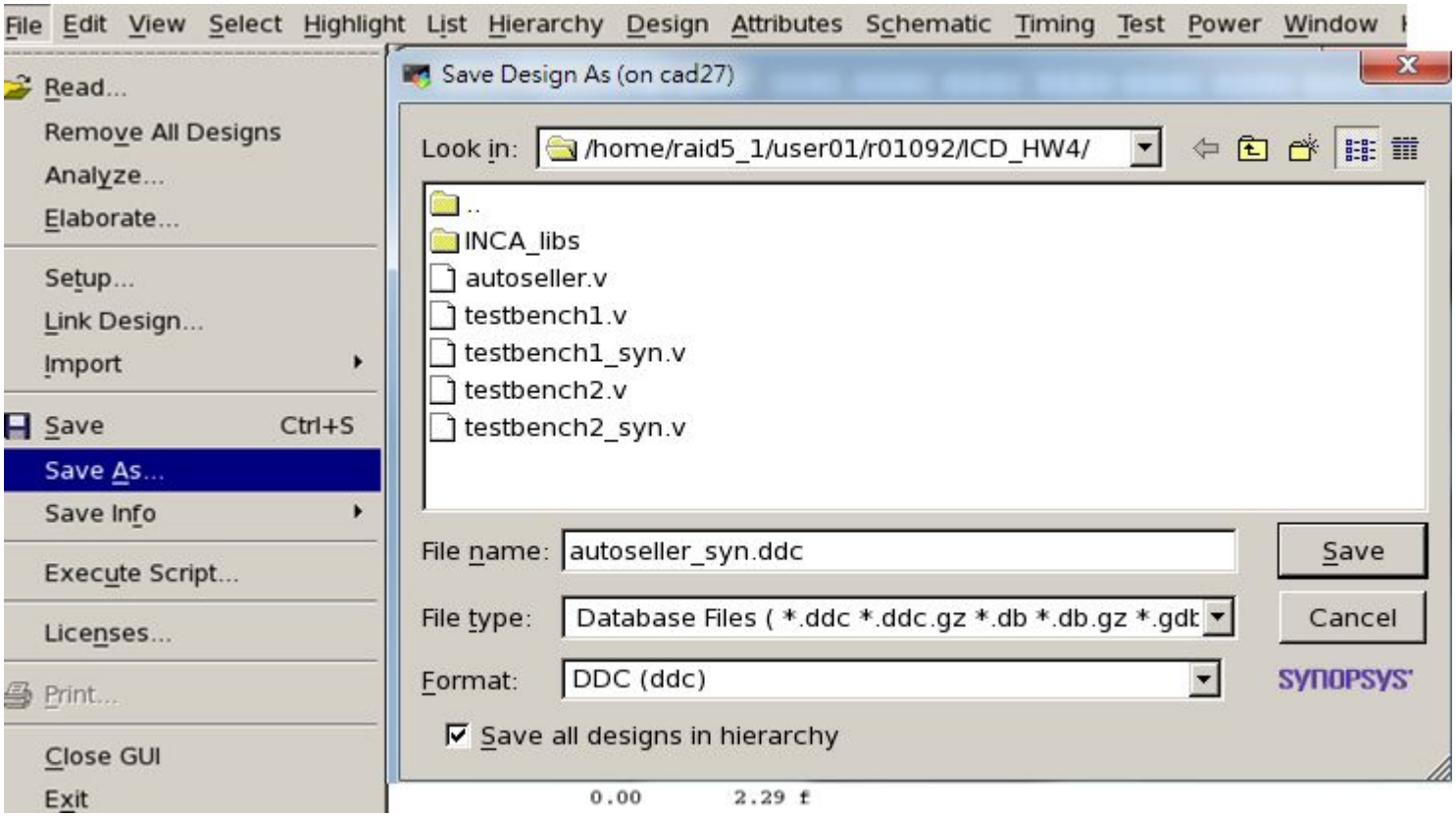
5. Write Essential Simulation Files

- **Write Gate Level Netlist**



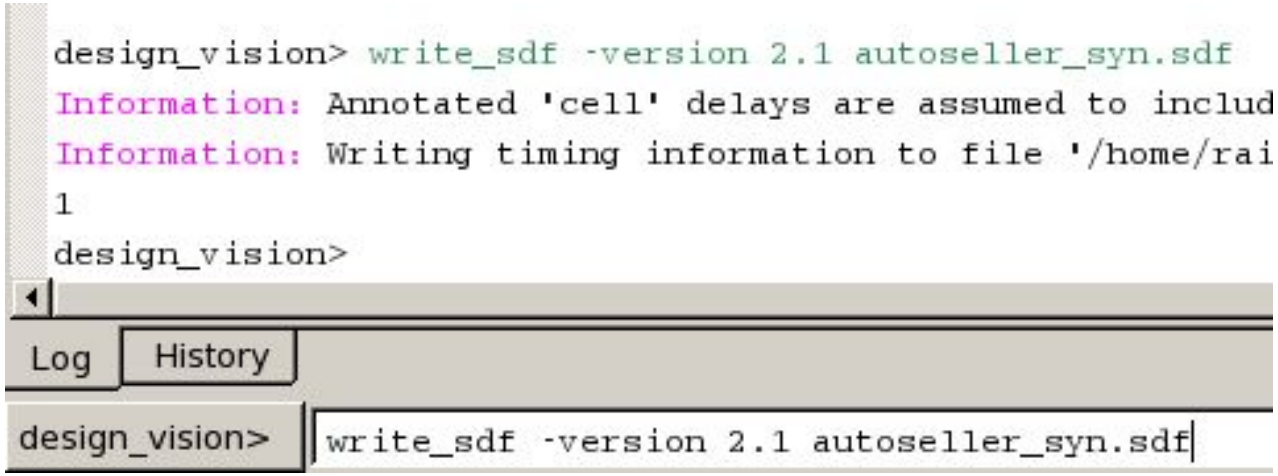
5. Write Essential Simulation Files

- **Write .ddc file.**
  - The .ddc file is for Design Vision Usage, you can restore the status of Design Vision



5. Write Essential Simulation Files

- **Write Standard Delay Format (SDF)**
  - The SDF file contains the delay information of standard cells.
  - For gate-level simulation, we can not simulate without the delay information of the standard cells.



Gate Level Simulation

- **Step 1**
  - Use $sdf_annotate in testbench.

```
initial begin
  $sdf_annotate("autoseller_syn.sdf", DUT);
  `ifdef FSDB
    $fsdbDumpfile("autoseller.fsdb");
    $fsdbDumpvars();
  `endif
end
```

- **Step 2**
  - Change the clock cycle in testbench if you need
    - Should be the same as the constraint set in DV

```
3   `define CYCLE 10
4   // modify if need
```

+define+FSDB

+define+FSDB+SDF

  - Run VCS with Gate-level netlist and tsmc13.v

```
% vcs <test bench> <gate-level netlist> <cell library> -full64 -R
+neg_tchk -debug_access+all +v2k
```

  - You may need to copy tsmc13.v to your own directory first.