

Computer-Aided VLSI System Design

Synthesis Lab 1-1: Design Vision / Design Compiler Synthesis Lab 1-2: Synopsys Memory Library Generation

Objectives:

In this lab, you will learn:

1. Basic concept about synthesis
2. How to use Synopsys Design Vision (GUI) / Design Compiler (text mode)
3. How to use Synopsys Design Compiler to generate the memory library file for Synopsys Design Compiler

Environment Setup:

1. Make sure you can run design compiler:

```
dc_shell -h
```

If it returned:

```
dc_shell: Command not found.
```

Go to edit ~/.cshrc, comment out line 90-92 like this:

```
#if (-f /usr/spring_soft/CIC/debussy.cshrc ) then
#  source /usr/spring_soft/CIC/debussy.cshrc
#endif
```

Then login again, or:

```
source ~/.cshrc
```

2. Make a working directory:

```
mkdir synthesis_lab
cd synthesis_lab
```

3. Copy synopsys_dc.setup to .synopsys_dc.setup into your working directory:

```
cp ~cvsd/CUR/Synthesis/Lab1/synopsys_dc.setup ./synopsys_dc.setup
```

Note: If you already have a .synopsys_dc.setup file and you do not want to overwrite it, you can add only the Library and tools related lines that you miss into your own .synopsys_dc.setup file.

Copy Files from CVSDTA Directory

1. Copy all the files into your work directory,

```
cp -r ~cvsd/CUR/Synthesis/Lab1/* .
```

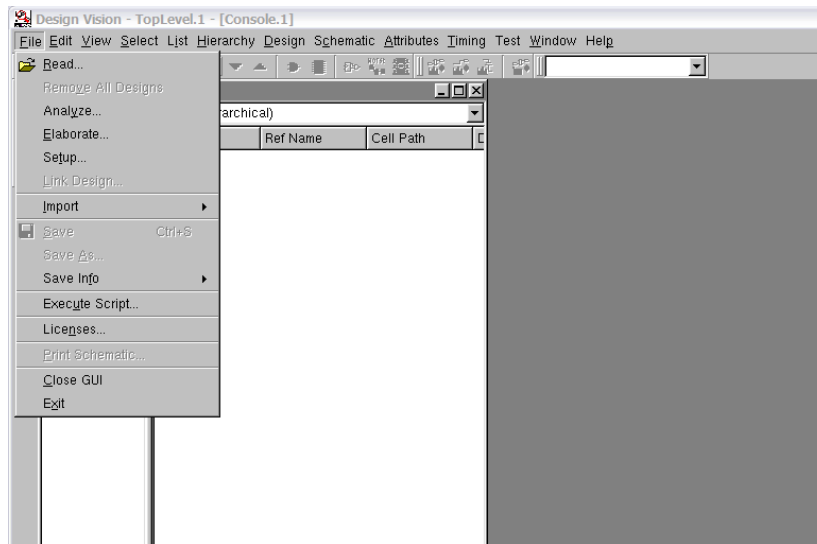
2. Check if you have these files

Filename	Description
.synopsys_dc.setup	Design compiler setting
Lab1_alu.v	The source ALU file
HSs18n_64x8_fast@-40C_syn.lib	The memory library under fast condition
HSs18n_64x8_fast@0C_syn.lib	The memory library under fast condition
HSs18n_64x8_slow_syn.lib	The memory library under slow condition
HSs18n_64x8_typical_syn.lib	The memory library under typical condition

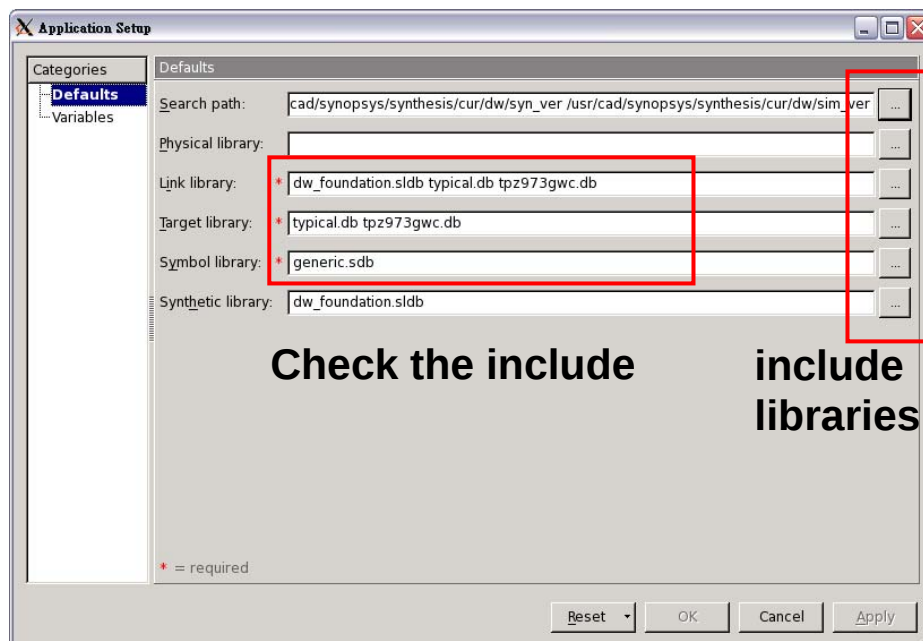
Lab1-1: Synopsys Design Vision

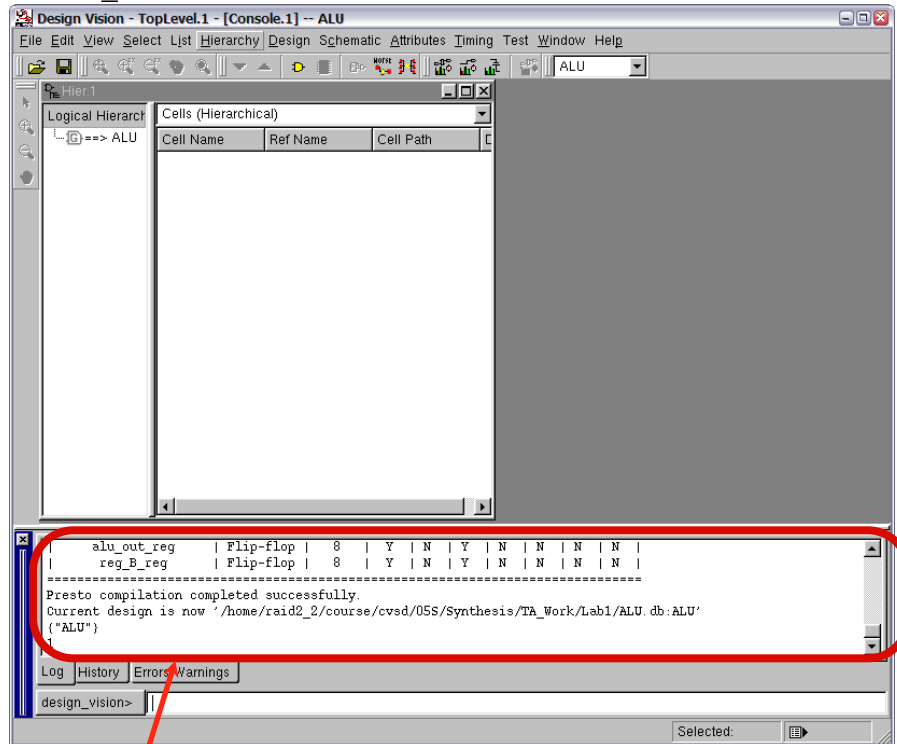
Read files

1. The function of *Lab1_alu.v* is exactly the same as Verilog course of Lab2. We will try to use this sample and practice Synopsys synthesis tool step by step. We will show the GUI example with figures and text-mode with blocked strings.
2. Build your working directory by yourself, copy all files and start up *design_vision* at working directory by using the **dv &** command. Also, you could invoke text mode by using **dc_shell** command or **Command line** at the bottom of the window.



3. Check the GUI environment setting "File -> Setup". If the "Line library" or "Target library" or "Symbol library" is empty, you have to check the *.synopsys_dc.setup* or use the GUI to choose the library files.

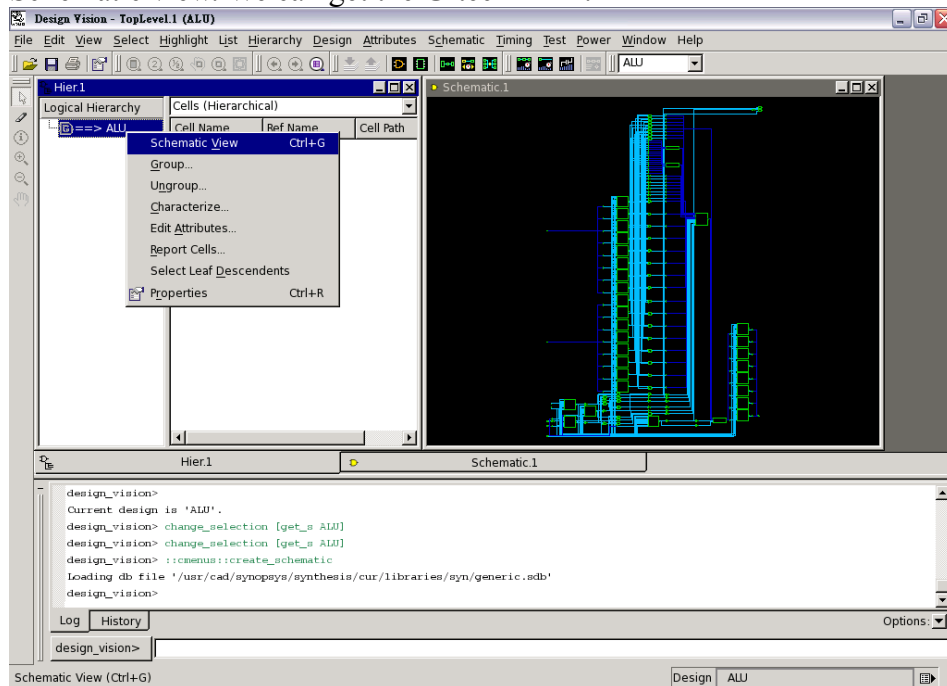


4. read *Lab1_alu.v*

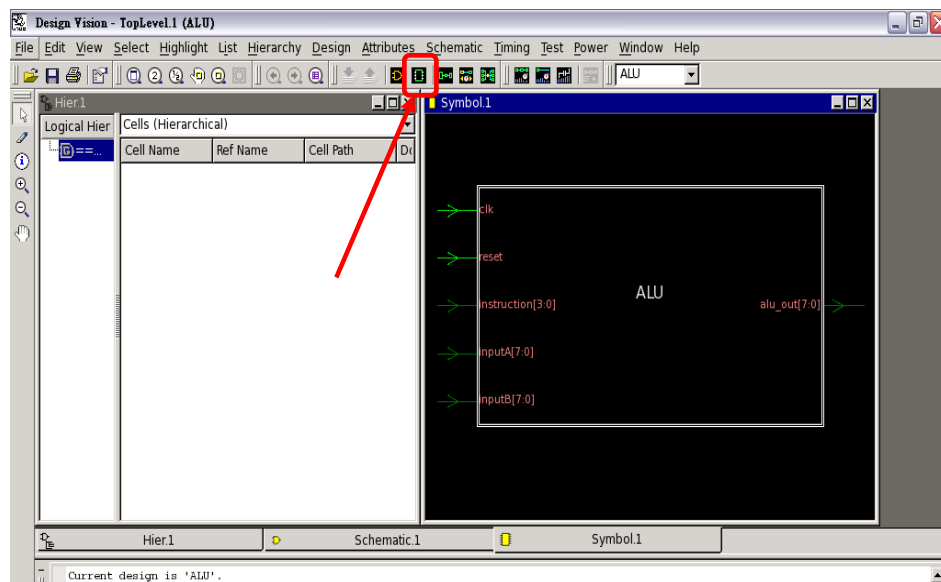
5. Check the log information. If any error or warning message, you have to fix it! After that, checking all the registers are flip-flop type. You have to modify your verilog code, if there is the **latch** in your circuit!

Register Name	Type	Width	Bus	MB	AR	AS	SR	SS	ST
reg_B_reg	Flip-flop	8	Y	N	Y	N	N	N	N
reg_A_reg	Flip-flop	8	Y	N	Y	N	N	N	N
reg_ins_reg	Flip-flop	4	Y	N	Y	N	N	N	N
alu_out_reg	Flip-flop	8	Y	N	Y	N	N	N	N

6. Synopsys Design analyzer will translate verilog code into G-tech model. Double click the icon “ALU”, and click the right button then choose Schematic view. We can get the G-tech MAP.

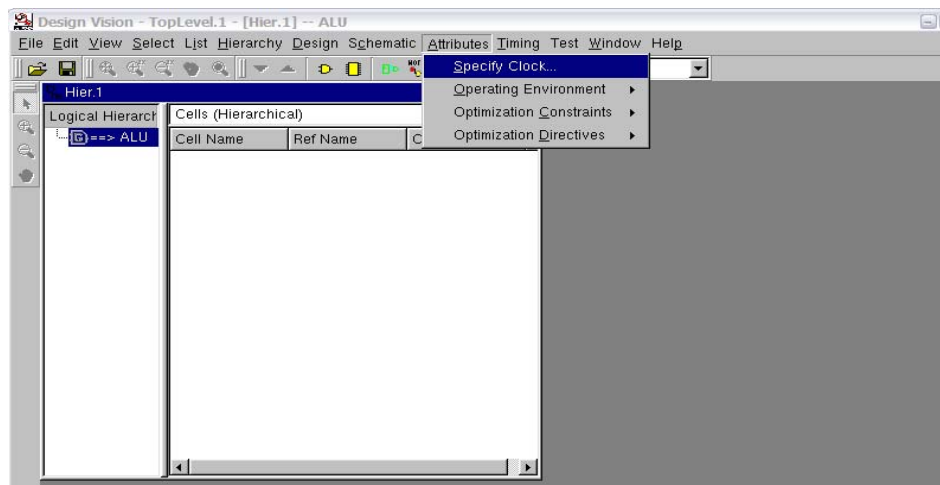


7. Or you can create a symbol view by click on the following symbol view button. The symbol view is as the right window.

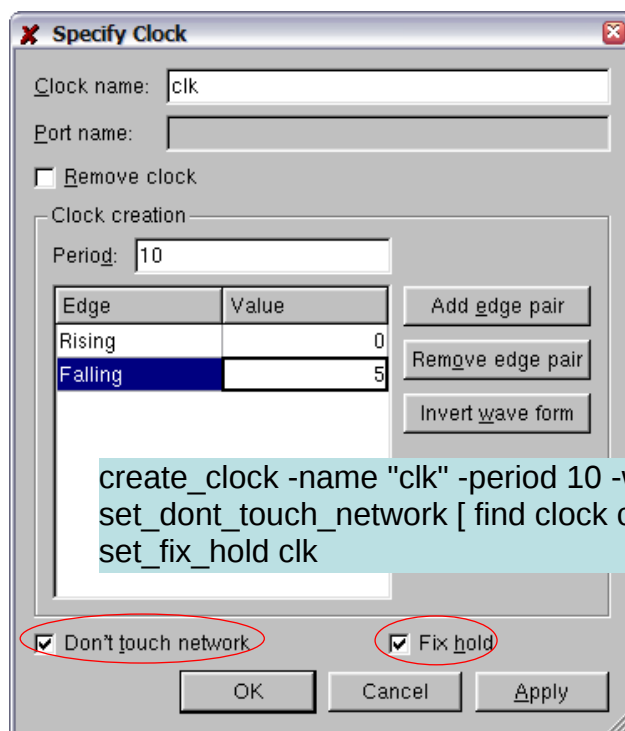


Set clock

- Back to the symbol view window. **Select the CLK(at Symbol View) port** and click “**A**tttributes”-“**S**pecify Clock”.



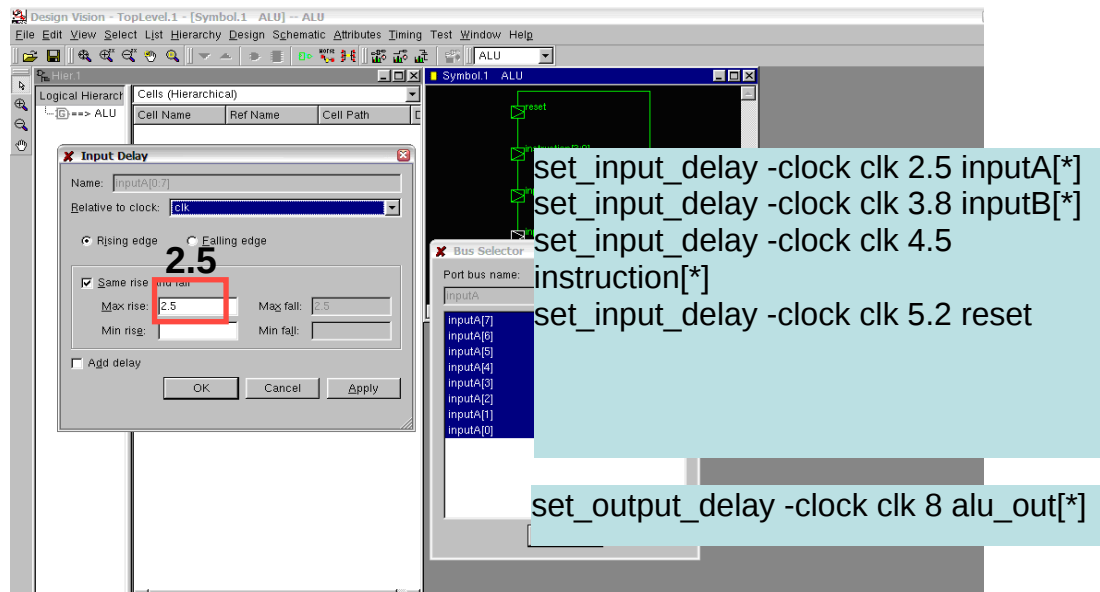
- Specify the clock as period 10ns. (100 MHz). Don't forget to select “**don't touch network**” and “**fix hold**”.



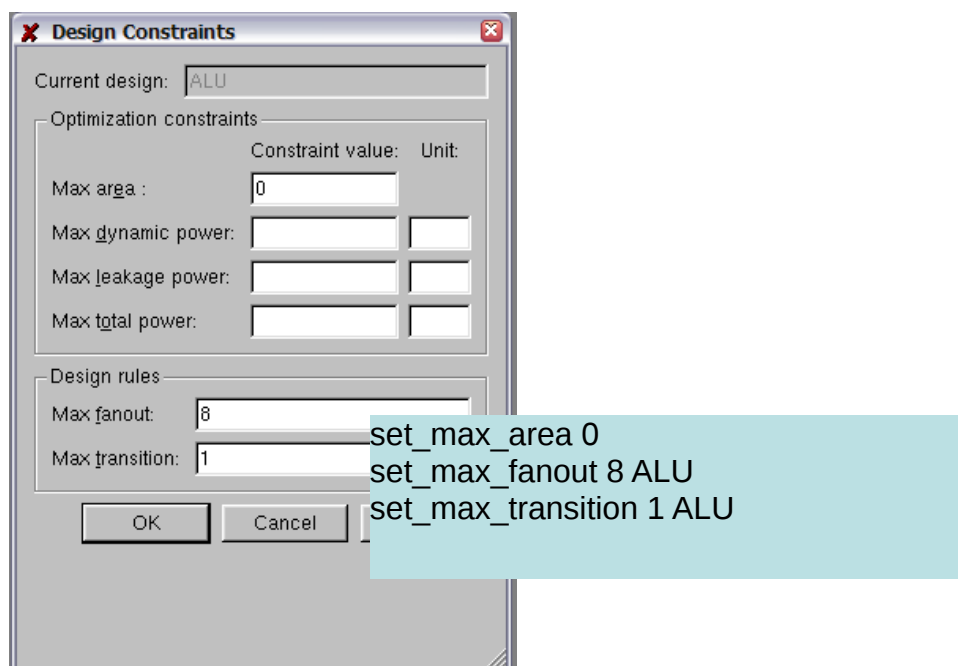
And then type in following command to change the wire load model:
set_operating_conditions "typical" -library "typical"
set_wire_load_model -name "ForQA" -library "typical"
set_wire_load_mode "segmented"

Set operating environment

10. Select “inputA” in the Symbol View and click “Attribute”-“operating environment”-“input delay”. Set 2.5ns input delay.



11. Set “input B” as 3.8ns input delay; “instruction” as 4.5ns input delay; “reset” as 5.2ns input delay.
12. By the same way, setup output delay 8ns on “alu_out”.
13. Click “Attribute”-“optimization Constraints”-“Design constraints”. Set max area is 0. Max fan-out is 8. max transition is 1. The error message in GUI mode could be ignored. And input following command:
set_boundary_optimization ""
set_fix_multiple_port_nets -all -buffer_constant

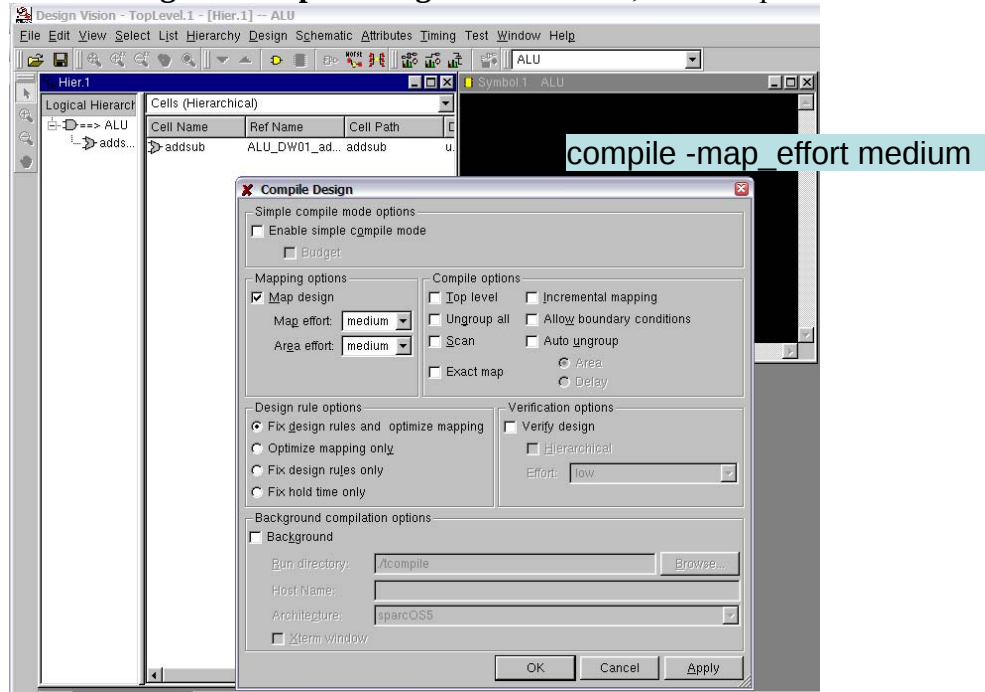


Check Design

- Click “**Design**”-”**Check Design**”. Click “OK”. If any warning or error messages, there are error at above steps. Redo these steps!

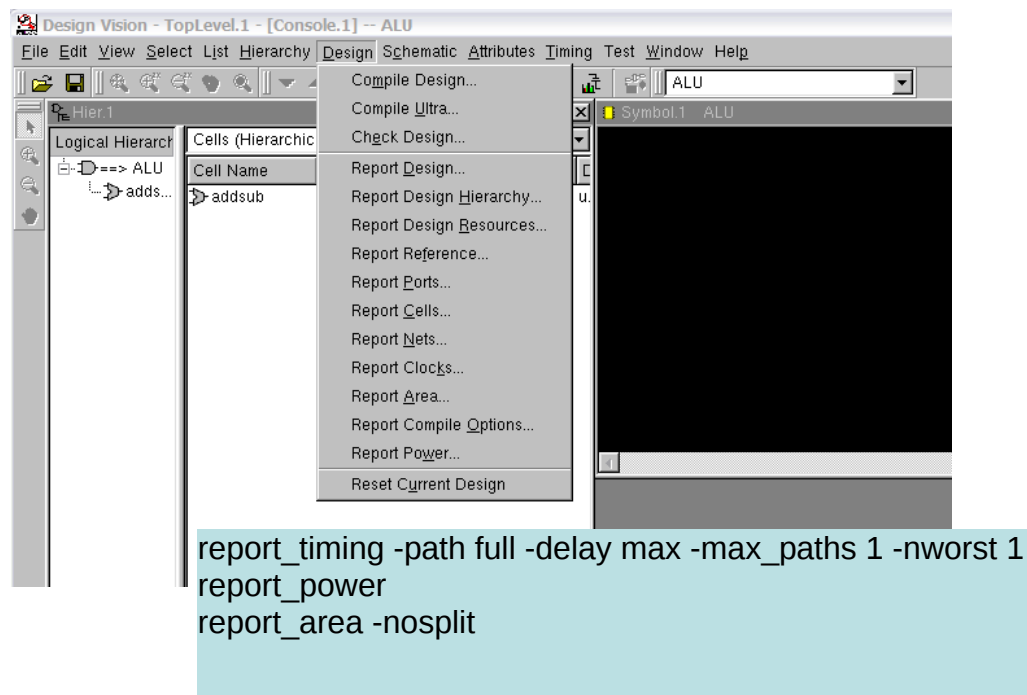
Design Optimization

- Click “**Design**”-”**Compile Design**”. Click “OK”, start to optimize ALU.



Report

- Few seconds later, We will get our gate level circuit. We must to check our circuit met our conditions or not at first.



17. Try to select different report item and check the report. For example:
 Check timing, use “Timing”=>“Report Timing Paths”; check power, use
 “Design”=>“Report Power”; check area, use “Design”=>“Report Area”

=====

== The following is timing report.

=====

Report : timing
 -path full
 -delay max
 -max_paths 1
 -sort_by group

Design : ALU
 Version: B-2008.09-SP2
 Date : Tue Oct 20 21:52:15 2009

Operating Conditions: typical Library: typical
 Wire Load Model Mode: segmented

Startpoint: alu_out_reg[0]
 (rising edge-triggered flip-flop clocked by clk)
 Endpoint: alu_out[0] (output port clocked by clk)
 Path Group: clk
 Path Type: max

Des/Clust/Port	Wire Load Model	Library
ALU	ForQA	typical
Point	Incr	Path
clock clk (rise edge)	0.00	0.00
clock network delay (ideal)	0.00	0.00
alu_out_reg[0]/CK (DFFRHQX1)	0.00	0.00 r
alu_out_reg[0]/Q (DFFRHQX1)	0.19	0.19 r
alu_out[0] (out)	0.00	0.19 r
data arrival time		0.19
clock clk (rise edge)	10.00	10.00
clock network delay (ideal)	0.00	10.00
output external delay	-8.00	2.00
data required time		2.00
data required time		2.00
data arrival time		-0.19
slack (MET)		1.81

Slack is Positive!!

=====

== The following is power report.

=====

Report : power
 -analysis_effort low

Design : ALU
 Version: B-2008.09-SP2
 Date : Tue Oct 20 21:53:46 2009

Library(s) Used:

```

    typical (File:
/home/raid1_1/cic/CBDK018_TSMC_Artisan/CIC/SynopsysDC/typical.
db)

```

```

Operating Conditions: typical  Library: typical
Wire Load Model Mode: segmented

```

Design	Wire Load Model	Library
ALU	ForQA	typical

```

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

```

```

    Cell Internal Power = 258.7260 uW    (92%)
    Net Switching Power = 20.9947 uW    (8%)
-----
Total Dynamic Power    = 279.7207 uW    (100%)

Cell Leakage Power     = 20.2175 nW

```

```

=====
== The following is area report.
=====

```

```

*****
Report : area
Design : ALU
Version: B-2008.09-SP2
Date   : Tue Oct 20 21:54:51 2009
*****

```

Library(s) Used:

```

    typical (File:
/home/raid1_1/cic/CBDK018_TSMC_Artisan/CIC/SynopsysDC/typical.db)

```

```

Number of ports:      30
Number of nets:       142
Number of cells:      104
Number of references: 14

Combinational area:   2195.424002
Noncombinational area: 1955.923218
Net Interconnect area: 18.000000

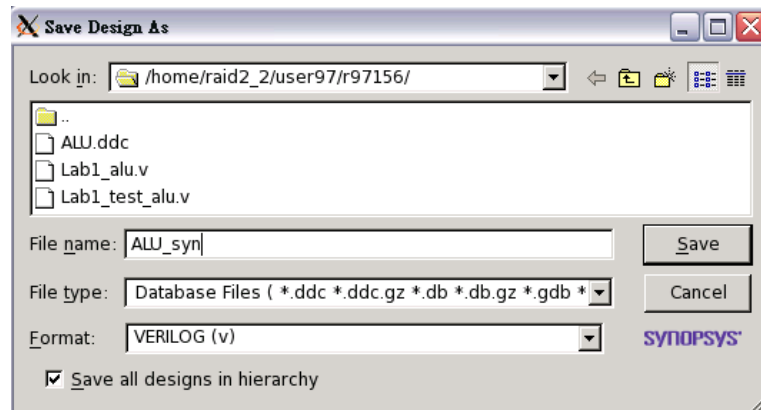
Total cell area:      4151.347220
Total area:           4169.347220

```

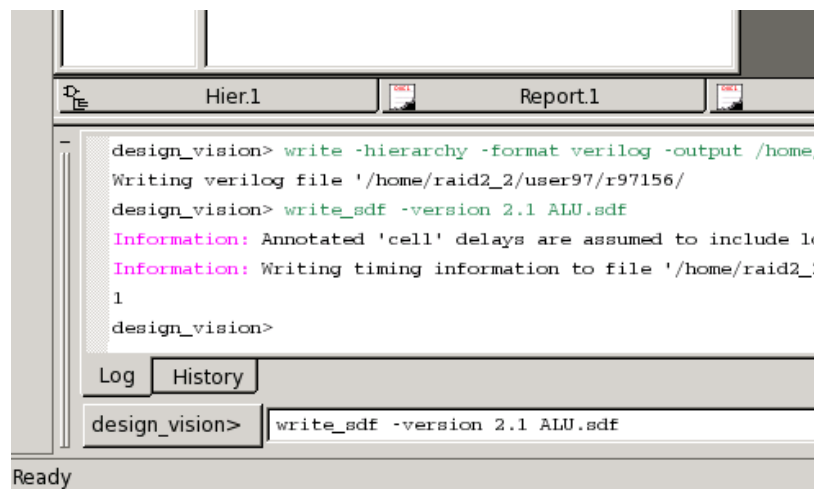
Notice at the last line "Slack". We know this circuit is safe at timing.

18. Repeat Step1. ~ Step 15. Try to change different settings and modify the constraints.

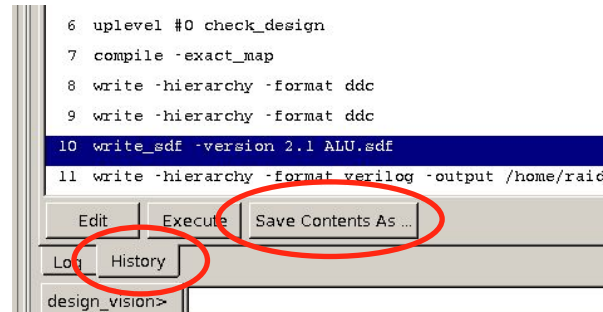
19. If the result is met your requirement, Synthesis is ending. Then, we must export the design to a file. Select “**File**”-“**Save**”(it will save the all the settings and results in **ALU.ddc**). You can also select “**File**”-“**Save As**” and choose “**DDC**” formate to save this design.
20. Save the gate level netlist in verilog formate. Select “**File**”-“**Save As**” , choose “**VERILOG**” formate with the File name “**ALU_syn**”, and click “**Save**”.



21. Finally, to save the timing information, you have to type the following commend in the command line: “**write_sdf -version 2.1 ALU.sdf**”. That will generate the timing information of this design.



22. You might use “**write_script > script_file**” command or save the comment history “**History**”-“**Save Contents As...**” to generate a script with the constraints you have made. After that, you could use “**include script_file**” command or “**File**”-“**Execute Script**” button to re-run all steps automatically.



23. For verilog gate-level simulation, you may add “\$sdf_annotate(sdf_file, testbench_module.ALU_instance);” in initial block in your test bench to use timing information for simulation.

Lab1-2: Synopsys Memory Library Generation**Environment Setup:**

1. Synopsys™ Design Compiler is not able to read the library file directly generated from the Memory Compiler. Please follow the below instructions to convert the .lib files to .db files that the Design Compiler can use.
2. We have gotten the TSMC 0.18μm 64 words * 8bit memory library files “HSs18n_64x8_fast@-40C_syn.lib”, “HSs18n_64x8_fast@0C_syn.lib”, ”HSs18n_64x8_slow_syn.lib”, and “HSs18n_64x8_typical_syn.lib” in the Verilog Lab2-4. Please check them whether they are in your working directory or not.

Compile the Memory Library:

3. Now we are going to compile the .lib files to .db files. Firstly, check your Synopsys environments are correctly setup.
4. Run **dc_shell -xg** under UNIX. And execute the following command in dc_shell:

```
dc_shell> read_lib NAME.lib
dc_shell> write_lib USER_LIB -output OUTPUT_FILE_NAME
```

You should fill the name of one of the three memory files (“HSs18n_64x8_fast@-40C_syn.lib”, “HSs18n_64x8_fast@0C_syn.lib”, ”HSs18n_64x8_slow_syn.lib”, and “HSs18n_64x8_typical_syn.lib) onto “NAME.lib” here. “USER_LIB” is the Library Name you defined while generating the “Synopsys library model(. lib)” in Memory Compiler.

As for HSs18n_64x8_fast_syn.lib, just type in following instructions:

```
dc_shell> read_lib HSs18n_64x8_typical_syn.lib
dc_shell> write_lib HSs18n_64x8 -output
HSs18n_64x8_typical_syn.db
```

The produced .db files are the synthesis models that Synopsys Design Compiler can use. Remember to add where you put the .db files in the “**search_path**” and add the .db file names in the “**link_library**” and “**target_library**” in your .synopsys_dc.setup before further synthesis of the RAM.

Checkpoints:

Please check with TAs before leaving this lab to make sure the following goals are accomplished and to get credits.

1. Show your synthesis result. Including synthesized gate level codes, sdf file, timing and area reports.
2. You have generated the memory db file. Please add them in to your .synopsys_dc.setup.

END of LAB

Creator:

1st Edition: Jazz Yang, 2002

2nd Edition: Lin Hsiung-mao, 2003
3rd Edition: Yu-Lin Chang, 2004, 2005
4th Edition: Yu-Lin Chang, 2006
5th Edition: Jui-Hsin Lai(Larry), 2008
6th Edition: Chih-Fan Lai, 2009
7th Edition: Chia-Wei Chang, 2010