IC设计基础实验

Lab1:Lab Environment

- Lab1:Basic Synthesis Design Flow
- 开始前:
  - 从桌面登入虚拟机，注意要求进入全屏状态，否则容易导致死机等未知情况！
  - Login as *student:student*

Lab1-1:Basic Synthesis Design Flow

1.右键新建一个终端：

```
$ lm
```

启动许可，需要一分钟左右，请等待；

2. 改变目录到SYNOPSYS/lab1

```
$ cd / SYNOPSYS/lab1
```

3.启动design_vision前，检查是否存在一个文件”.synopsys_dc.setup”（右键新建一个终端：）

```
$ ls –la .sysnopsys_dc.setup
$ more .sysnopsys_dc.setup
$ dv
```

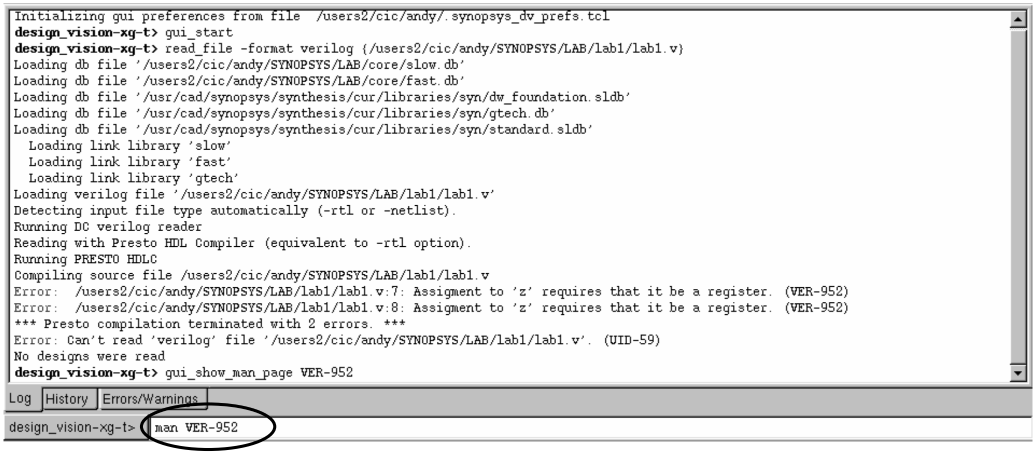
随后会启动一个图形界面，请稍候

4.读入”lab1.v”：使用主菜单“File/Read”读取“lab1.v”，存在什么问题？______________

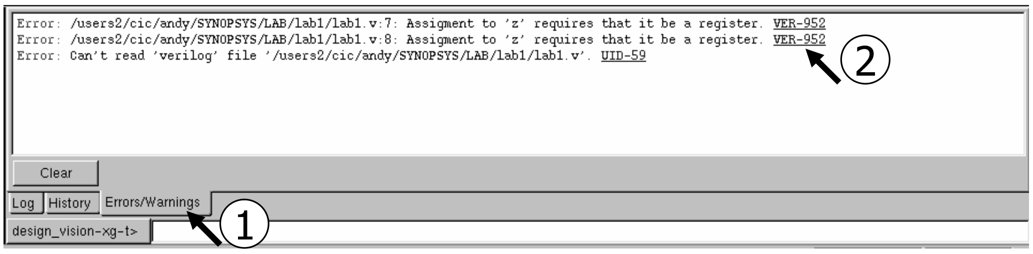
Find error

4. Use "*man error_no*" in dc_shell-xg-t

method 1


```
Initializing gui preferences from file  /users2/cic/andy/.synopsys_dv_prefs.tcl
design_vision-xg-t> gui_start
design_vision-xg-t> read_file -format verilog {/users2/cic/andy/SYNOPSYS/LAB/lab1/lab1.v}
Loading db file '/users2/cic/andy/SYNOPSYS/LAB/core/slow.db'
Loading db file '/users2/cic/andy/SYNOPSYS/LAB/core/fast.db'
Loading db file '/usr/cad/synopsys/synthesis/cur/libraries/syn/dw_foundation.sldb'
Loading db file '/usr/cad/synopsys/synthesis/cur/libraries/syn/gtech.db'
Loading db file '/usr/cad/synopsys/synthesis/cur/libraries/syn/standard.sldb'
  Loading link library 'slow'
  Loading link library 'fast'
  Loading link library 'gtech'
Loading verilog file '/users2/cic/andy/SYNOPSYS/LAB/lab1/lab1.v'
Detecting input file type automatically (-rtl or -netlist).
Running DC verilog reader
Reading with Presto HDL Compiler (equivalent to -rtl option).
Running PRESTO HDLC
Compiling source file /users2/cic/andy/SYNOPSYS/LAB/lab1/lab1.v
Error:  /users2/cic/andy/SYNOPSYS/LAB/lab1/lab1.v:7: Assigment to 'z' requires that it be a register.  (VER-952)
Error:  /users2/cic/andy/SYNOPSYS/LAB/lab1/lab1.v:8: Assigment to 'z' requires that it be a register.  (VER-952)
*** Presto compilation terminated with 2 errors. ***
Error: Can't read 'verilog' file '/users2/cic/andy/SYNOPSYS/LAB/lab1/lab1.v'. (UID-59)
No designs were read
design_vision-xg-t> gui_show_man_page VER-952
```

method 2

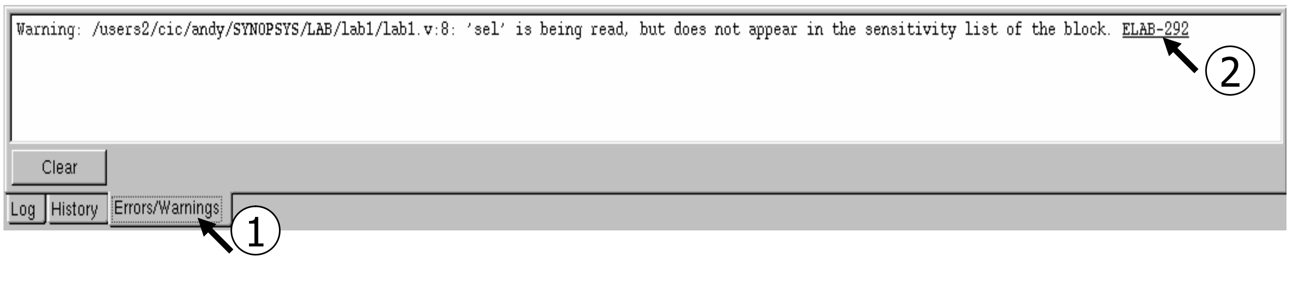

```
Error: /users2/cic/andy/SYNOPSYS/LAB/lab1/lab1.v:7: Assigment to 'z' requires that it be a register. VER-952
Error: /users2/cic/andy/SYNOPSYS/LAB/lab1/lab1.v:8: Assigment to 'z' requires that it be a register. VER-952
Error: Can't read 'verilog' file '/users2/cic/andy/SYNOPSYS/LAB/lab1/lab1.v'. UID-59
```

Fix the error

5. Modify the "***lab1.v***" to fix the error (You can use ***vi*** or other text editor)

6. Read the file "***lab1.v***" again to see if there still have any error message or warning message. If exists, what is the error or warning # ?
   ____

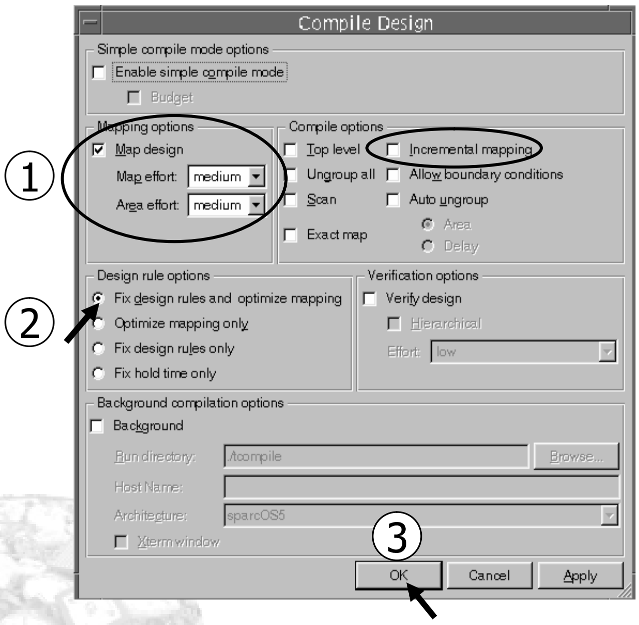
7. Look up this error and modify the "lab1.v" again.


```
Warning: /users2/cic/andy/SYNOPSYS/LAB/lab1/lab1.v:8: 'sel' is being read, but does not appear in the sensitivity list of the block. ELAB-292
```

Synthesis Your Design

8. Compile the design#

   Use the#*dv* menu bar#"***Design/Compile Design***#" to synthesize your# design. After synthesis, look at the hierarchy view, what happened? # One more#"***lab1*_DW01_add_0**" appears. Why does this# lab1_DW01_add_0 appear?#______________________


Compile Design

Simple compile mode options
- [ ] Enable simple compile mode
- [ ] Budget

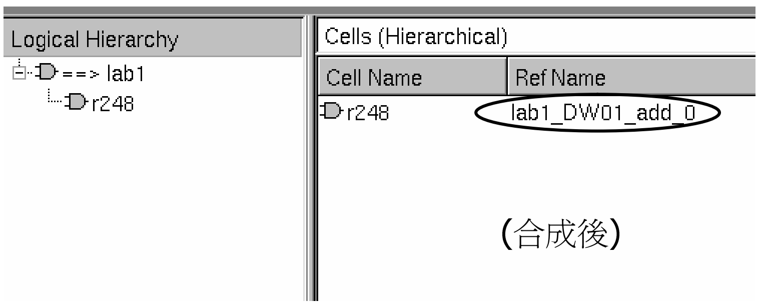
Mapping options
- [✓] Map design
- Map effort: medium
- Area effort: medium

Compile options
- [ ] Top level
- [ ] Ungroup all
- [ ] Scan
- [ ] Exact map
- [ ] Incremental mapping
- [ ] Allow boundary conditions
- [ ] Auto ungroup
- ● Area
- ○ Delay

Design rule options
- ● Fix design rules and optimize mapping
- ○ Optimize mapping only
- ○ Fix design rules only
- ○ Fix hold time only

Verification options
- [ ] Verify design
- [ ] Hierarchical
- Effort: low

Background compilation options
- [ ] Background
- Run directory: /compile
- Host Name:
- Architecture: sparcOS5
- [ ] Xterm window


Logical Hierarchy
- ==> lab1
  - r248

Cells (Hierarchical)

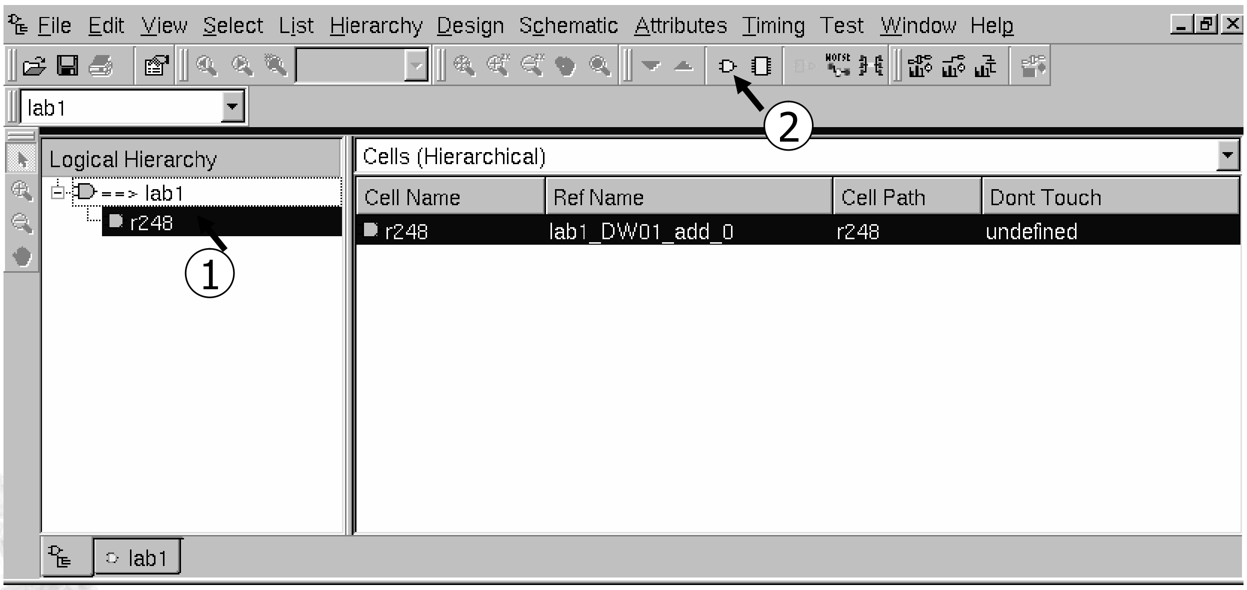
Cell Name	Ref Name
r248	lab1_DW01_add_0

(合成後)

Create Design Schematic

9. "**Create Design Schematic**" to see the result after synthesis. How# many adders are there used after synthesizing? __________#

10. Choose the Ref named#"***lab1_DW01_add_0***",#"**Create Design# Schematic**" to see the structure of this adder. What type of the adder# is synthesized#– ***cla*** or#***ripple*** or other form? ________#



File Edit View Select List Hierarchy Design Schematic Attributes Timing Test Window Help
lab1
①
②

Logical Hierarchy
- ==> lab1
  - r248

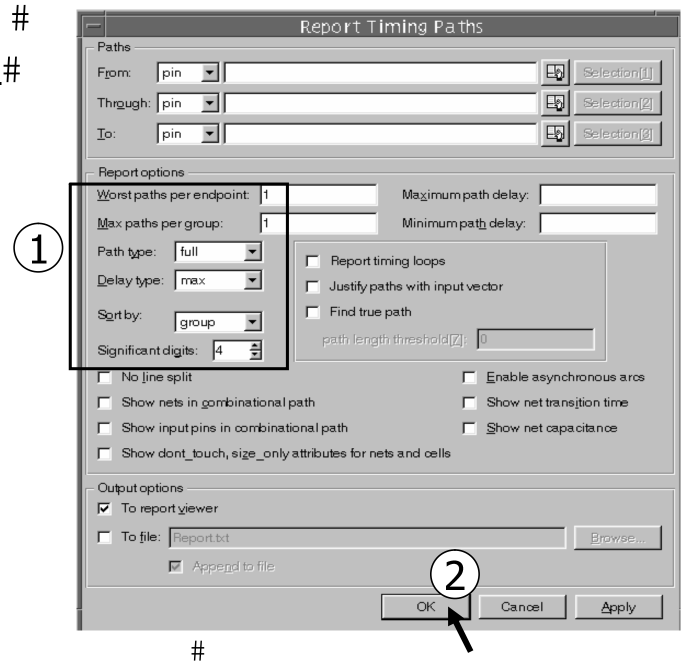
Cells (Hierarchical)

Cell Name	Ref Name	Cell Path	Dont Touch
r248	lab1_DW01_add_0	r248	undefined

Report#– timing & area & resources

11. To select the#"*lab1*". Use#"*Design/Report Design Resources*" to# see what type of adder is used by design compiler? ________

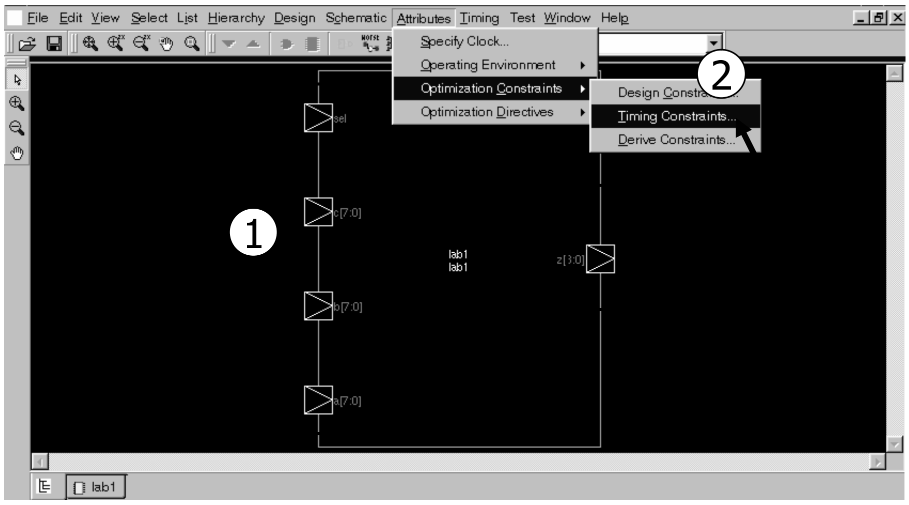
12. Use the#*dv* menu bar#"*Design/Report Area#* and#"*Timing/Report# Timing#* to see the#*timing & area*.
    area = ________ , #
    timing= ________#

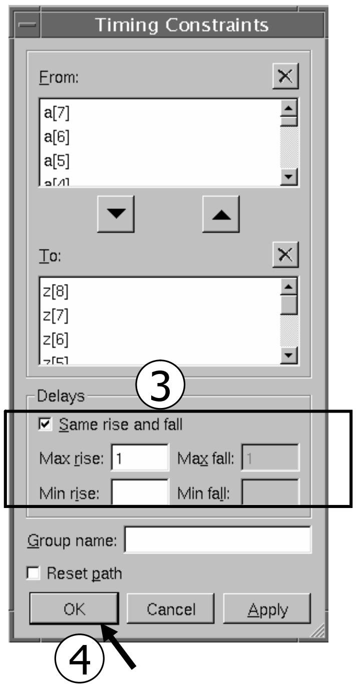


Worst paths per endpoint: 1
Max paths per group: 1
Path type: full
Delay type: max
Sort by: group
Significant digits: 4
Maximum path delay:
Minimum path delay:
☐ Report timing loops
☐ Justify paths with input vector
☐ Find true path
path length threshold[Z]: 0
☐ No line split
☐ Show nets in combinational path
☐ Show input pins in combinational path
☐ Show dont_touch, size_only attributes for nets and cells
☐ Enable asynchronous arcs
☐ Show net transition time
☐ Show net capacitance
☑ To report viewer
☐ To file: Report.txt
☑ Append to file

Set Timing Constraints

13. In the#"*Symbol View*", choose all input and output ports and in the#*dv* menu bar, choose#"***Attributes/Optimization Constraints/Timing# Constraints#"*** ➔ Set the#"**Maximum Delay"** to 1 , then **repeat the# step 8-12**



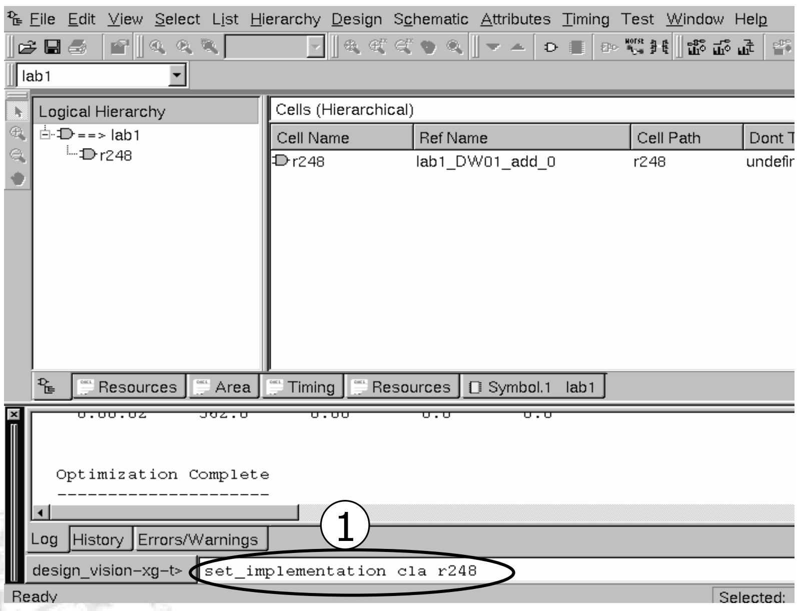


☑ Same rise and fall
Max rise: 1 Max fall: 1
Min rise: Min fall:
☐ Reset path

Change Different#DesignWare

14.#"**Create Design Schematic**" of adder to see the structure of this# adder. What is this adder synthesized#– ***cla*** or#***ripple*** or#***bk*** or#**pparch** or other implementation form? _____________. What causes this?# ____#

15. How can I change this adder to#cla structure? ___________________#

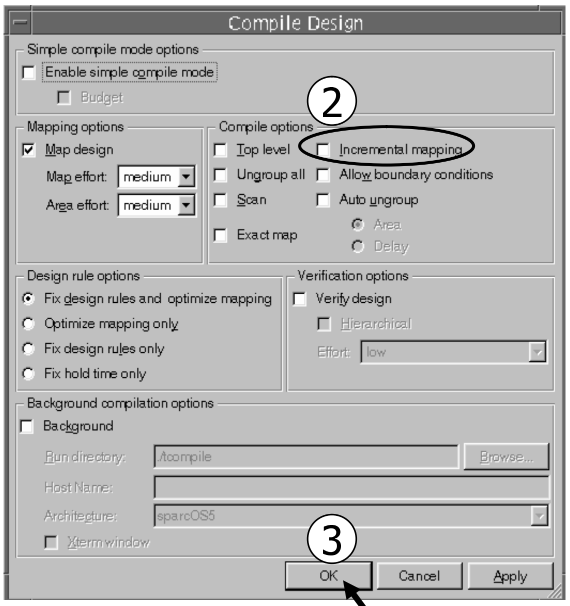


Cells (Hierarchical)

Cell Name	Ref Name	Cell Path	Dont T
r248	lab1_DW01_add_0	r248	undefir

```
Optimization Complete
---------------------
```

```
design_vision-xg-t> set_implementation cla r248
```



Simple compile mode options
- ☐ Enable simple compile mode
- ☐ Budget

Mapping options
- ☑ Map design
- Map effort: medium
- Area effort: medium

Compile options
- ☐ Top level
- ☐ Incremental mapping
- ☐ Ungroup all
- ☐ Allow boundary conditions
- ☐ Scan
- ☐ Auto ungroup
- ● Area
- ○ Delay
- ☐ Exact map

Design rule options
- ● Fix design rules and optimize mapping
- ○ Optimize mapping only
- ○ Fix design rules only
- ○ Fix hold time only

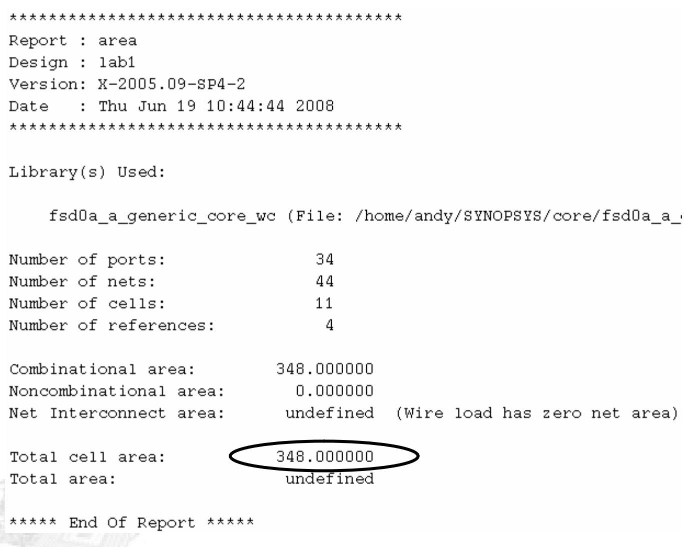
Verification options
- ☐ Verify design
- ☐ Hierarchical
- Effort: low

Background compilation options
- ☐ Background
- Run directory: ./compile
- Host Name:
- Architecture: sparcOS5
- ☐ Xterm window

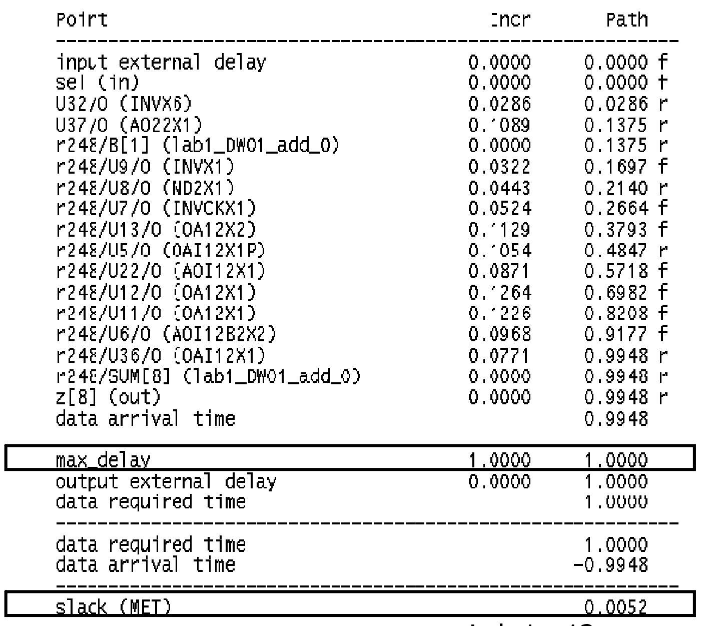
Report#– timing & area & resources

16."***Design/Report Design Resources***" to see what type of adder is# used by DC? ___________

17. "***Design/Report Area#&#Timing/Report Timing#***" , is there any# difference with#***step12?*** area=________, timing= ______



```
****************************************
Report : area
Design : lab1
Version: X-2005.09-SP4-2
Date   : Thu Jun 19 10:44:44 2008
****************************************

Library(s) Used:

    fsd0a_a_generic_core_wc (File: /home/andy/SYNOPSYS/core/fsd0a_a_

Number of ports:               34
Number of nets:                44
Number of cells:               11
Number of references:           4

Combinational area:       348.000000
Noncombinational area:      0.000000
Net Interconnect area:      undefined  (Wire load has zero net area)

Total cell area:          348.000000
Total area:                 undefined

***** End Of Report *****
```



```
Point                                    Incr       Path
---------------------------------------------------------
input external delay                     0.0000     0.0000 f
sel (in)                                 0.0000     0.0000 f
U32/O (INVX6)                            0.0286     0.0286 r
U37/O (AO22X1)                           0.1089     0.1375 r
r248/B[1] (lab1_DW01_add_0)              0.0000     0.1375 r
r248/U9/O (INVX1)                        0.0322     0.1697 f
r248/U8/O (ND2X1)                        0.0443     0.2140 r
r248/U7/O (INVCKX1)                      0.0524     0.2664 f
r248/U13/O (OA12X2)                      0.1129     0.3793 f
r248/U5/O (OAI12X1P)                     0.1054     0.4847 r
r248/U22/O (AOI12X1)                     0.0871     0.5718 f
r248/U12/O (OA12X1)                      0.1264     0.6982 f
r248/U11/O (OA12X1)                      0.1226     0.8208 f
r248/U6/O (AOI12B2X2)                    0.0968     0.9177 f
r248/U36/O (OAI12X1)                     0.0771     0.9948 r
r248/SUM[8] (lab1_DW01_add_0)            0.0000     0.9948 r
z[8] (out)                               0.0000     0.9948 r
data arrival time                                   0.9948

max_delay                                1.0000     1.0000
output external delay                    0.0000     1.0000
data required time                                  1.0000
---------------------------------------------------------
data required time                                  1.0000
data arrival time                                  -0.9948
---------------------------------------------------------
slack (MET)                                         0.0052
```

Lab 1 Answers / Solutions

3. Error Message: VER-952
5. 在output [8:0] z; 之後多加一行 reg [8:0] z; (或 output reg [8:0] z; 比較快!)
6. Warning Message: ELAB-292
7. 修改always @(a or b or c) begin 這行改成always @(a or b or c or#sel) begin
8. 因爲合成之後DC會將RTL Code內的"+",#用實際的DesignWare Library取代之.
9. 1#個 (因爲有用到Resource Sharing技巧)
10. 從schematic view觀察,#可猜測是Ripple adder
11. 用report_design_resource來觀察的確是 rpl
12. Area= 321 um^2, Timing= 1.1936 ns
14. pparch. 因爲我們剛剛有針對Combination電路設定Timing Constraints, Tool幫我們隨意挑選一個可以滿足1ns以內完成計算的加法器.
15. set_implementation cla r248 (cell name不見得一定是r248 僅供參考!)
    設定完之後需要再作一次合成 => compile
16. cla
17. Area= 348 um^2, Timing= 0.9948 ns.

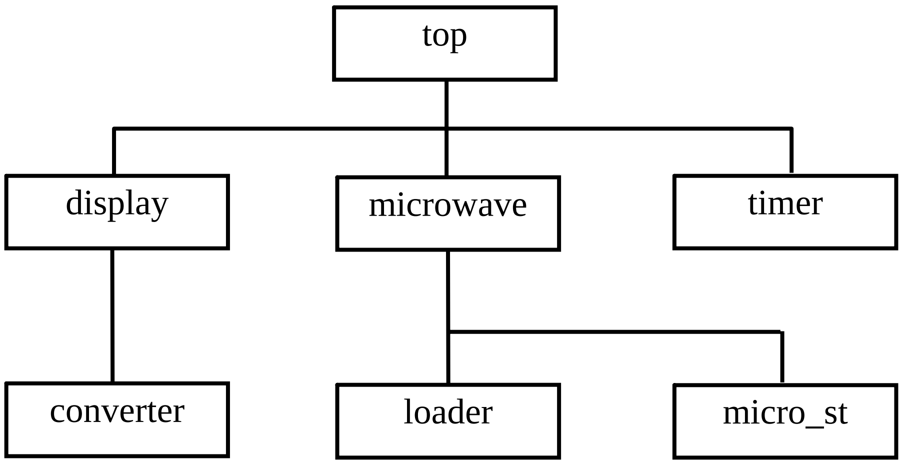
Lab2-1: Block Level Design (UMC90)

- Setting Design Constraints & Compile Design & Report & Analysis

These labs take approximately 1 hour to complete

Introduction – Microwave Timer System

- The top of the hierarchical design consists of 3 blocks, ***microwave***, ***timer*** and ***display***. ***microwave*** contains a state machine, **micro_st**, which generates the control signal, and a **loader** that loads the time we want to cook, **cook_time[15:0]**, to the **timer**. **timer** will decrease the **cook_time** per second, and **display** unit uses four 7 segments LEDs to display the cook time. As the cook time reach zero, these LED will display "donE" to inform the completion of cooking.



Microwave Timer Specification

- The input and output of the top design is described below.
  - **clk** is the synchronous clock of this design.
  - **reset** is used to reset the microwave timer. If reset is zero, the design is reset to IDLE state. Until it changed to 1, then the timer start work.
  - **test** pin is used to test if LEDs are OK. When test is 1, LED will display "8888".
  - **cook_time** is the time we want to cook.
  - **set_time** is used to set the cook time. When set_time is 1, cook_time is load into the timer.
  - **start_cook** indicates starting of cook. When start_cook is changed to 1, cooking starts, and the cook time display is decreased one per second as time passes by.
  - **min_msb_led, min_lsb_led, sec_msb_led** and **sec_lsb_led** are the outputs of BCD-to-7-segment converter, which are used to control the LEDs.

Getting Start

本**Lab**之**Script**是**"script.tcl"**
不喜歡打字的同學可以直接用**!**

1. Enter the directory **SYNOPSYS/lab2/lab2-1**,
   unix% cd ~/SYNOPSYS/lab2/lab2-1
   unix% ls –al

2. Check the contents of **.synopsys_dc.setup**.
   unix% more .synopsys_dc.setup

3. Invoke the Design Vision XG mode
   unix% dv

4. Check the search path, link library, target library and symbol library. We will use **fsd0a_a_generic_core_wc.db.db** & **fod0a_b33_t33_generic_io_wc.db** as link library and target library, **generic.sdb** as symbol library.

   File -> Setup
   If it is ok, **Cancel**
   Compare it with the content of .synopsys_dc.setup
   more .synopsys_dc.setup

Read Design

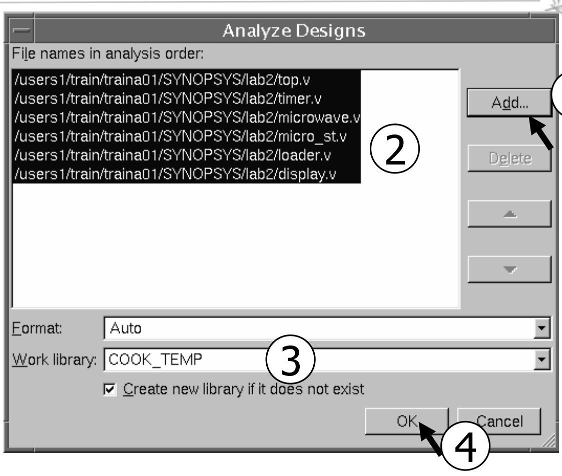
5. Read files
   File#> Read
   File type: All Files (.*)
   Click on#**converter.pla**
   Open

切記: 副檔名格式不一樣的檔案, 在DC讀檔案時, 不可以一起讀取! 相同格式的檔案如Verilog 可以全部一起讀取! 沒有個數限制~

6. To read the other#verilog files again.
   File#> Analyze
   Add…
   Use#*left key* click on#**display.v**, then# additional#"*Ctrl#" key* click on#**loader.v**,# **microwave.v**,#**micro_st.v**,#**timer.v** and# **top.v**
   Select
   Enter#"**COOK_TEMP**" Library
   OK

   File#> Elaborate
   Choose#"**COOK_TEMP**" Library
   Design: **top(verilog)**
   OK

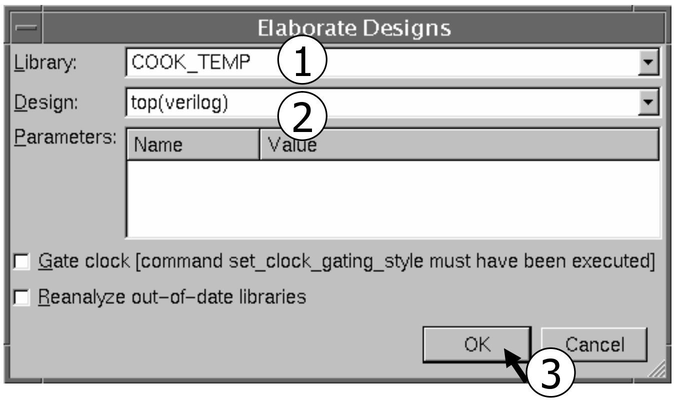
注意: Step 6 做完時, 請打link指令看有無錯誤! link指令可以幫忙檢查file&Lib是否正確! 請務必修正到正確爲止再進行Step7.


Analyze Designs

File names in analysis order:

```
/users1/train/traina01/SYNOPSYS/lab2/top.v
/users1/train/traina01/SYNOPSYS/lab2/timer.v
/users1/train/traina01/SYNOPSYS/lab2/microwave.v
/users1/train/traina01/SYNOPSYS/lab2/micro_st.v
/users1/train/traina01/SYNOPSYS/lab2/loader.v
/users1/train/traina01/SYNOPSYS/lab2/display.v
```

Format: Auto
Work library: COOK_TEMP
☑ Create new library if it does not exist


Elaborate Designs

Library: COOK_TEMP
Design: top(verilog)
Parameters:

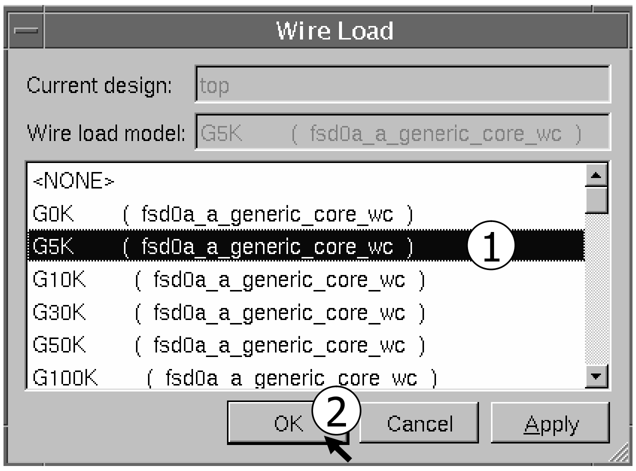
Name	Value

☐ Gate clock [command set_clock_gating_style must have been executed]
☐ Reanalyze out-of-date libraries

Setting Operating Condition & Wire Load

7. To set Operating Conditional
dc_shell-xt-t> set_operating_conditions -max WCCOM #min BCCOM

8. Attributes#➔ Operating Environment#➔ Wire Lode
Click on#**G5K**
OK


Wire Load

Current design: top
Wire load model: G5K (fsd0a_a_generic_core_wc)

```
<NONE>
G0K    ( fsd0a_a_generic_core_wc )
G5K    ( fsd0a_a_generic_core_wc )
G10K   ( fsd0a_a_generic_core_wc )
G30K   ( fsd0a_a_generic_core_wc )
G50K   ( fsd0a_a_generic_core_wc )
G100K  ( fsd0a_a_generic_core_wc )
```

dc_shell-xg-t> set_wire_load_mode top ③

Setting Clock Constraints / Driving Strength

9. To create clock signal, and set it's attributes in#dc_shell-xt command# line. (In DC 2005.09-sp4, you can't create clock by DC GUI.)

```
dc_shell-xg-t>#create_clock  -name #clk  -period  4   [get_ports clk]
dc_shell-xg-t>#set_dont_touch_network                 [get_clocks clk]
dc_shell-xg-t>#set_fix_hold                           [get_clocks clk]
dc_shell-xg-t>#set_clock_uncertainty             0.1  [get_clocks clk]
dc_shell-xg-t>#set_clock_latency                   1  [get_clocks clk]
```

10. View the symbol view of top

11. Setting input driving strength for#clk port

```
dc_shell-xg-t>#set_driving_cell  -library#fsd0a_a_generic_core_wc  \
               -lib_cell BUFX4  #pin  {O}  [get_ports clk]
```

12. Setting input driving strength for#**all input port** *except* **clk**

```
dc_shell-xg-t>#set_driving_cell -library#fsd0a_a_generic_core_wc  \
               -lib_cell DFFX1 #-pin  {Q} [remove_from_collection  \
                [all_inputs]  [get_ports clk]]
```

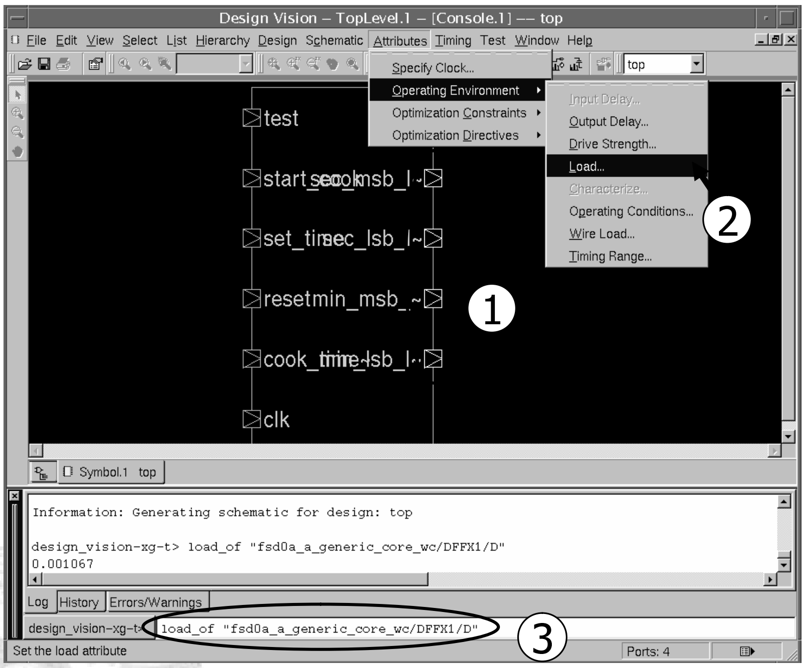
不喜歡打字的同學可以將**script.tcl**檔案打開,#複製**step9~step12**的指令後,#直接執行**!**

Setting Output Capacitance Loading

13. Select all output ports by drag the left key

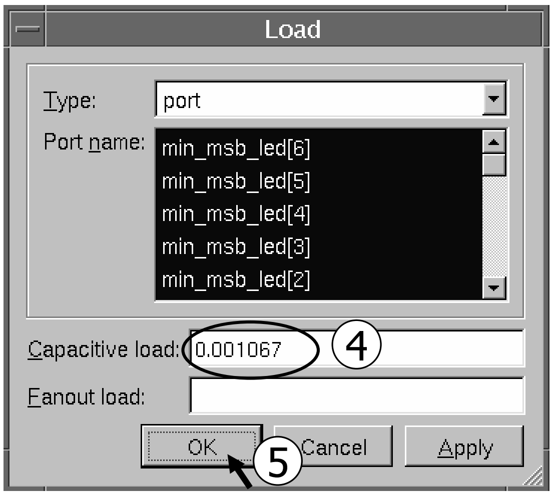
Attributes#➔ Operating Environment#➔ Load

Enter#**load_of "fsd0a_a_generic_core_wc /DFFX1/D"**



```
Information: Generating schematic for design: top

design_vision-xg-t> load_of "fsd0a_a_generic_core_wc/DFFX1/D"
0.001067
```

```
design_vision-xg-t> load_of "fsd0a_a_generic_core_wc/DFFX1/D"
```



Type: port

Port name:
min_msb_led[6]
min_msb_led[5]
min_msb_led[4]
min_msb_led[3]
min_msb_led[2]

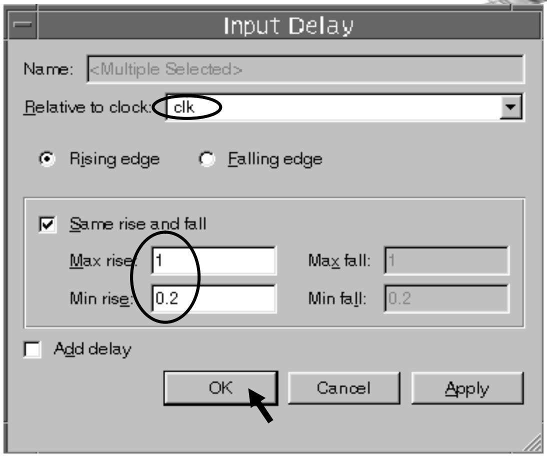
Capacitive load: 0.001067

Fanout load:

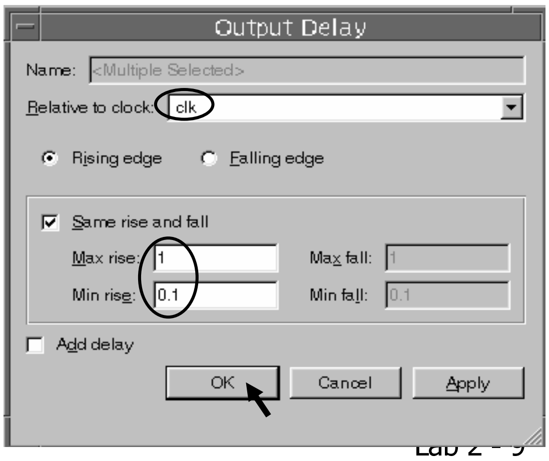
Setting Input /Output Delay#

14. Setting input delay
    Select#**all input ports**
    Attributes#➔ Operating# Environment#➔ Input Delay
    Relative To Clock is set#**clk**
    Max Rise & Fall as #**1**
    Min Rise & Fall as #**0.2**
    OK

15. Select#**all the output ports**
    Attributes#➔ Operating# Environment#➔ Output Delay
    Relative To Clock is set to#**clk**
    Max Rise and Fall as#**1**
    Min Rise and Fall as #**0.1**
    OK


Input Delay

Name: <Multiple Selected>
Relative to clock: clk
[✓ Rising edge] Falling edge
[✓ Same rise and fall]
Max rise: 1 Max fall: 1
Min rise: 0.2 Min fall: 0.2
Add delay


Output Delay

Name: <Multiple Selected>
Relative to clock: clk
[✓ Rising edge] Falling edge
[✓ Same rise and fall]
Max rise: 1 Max fall: 1
Min rise: 0.1 Min fall: 0.1
Add delay

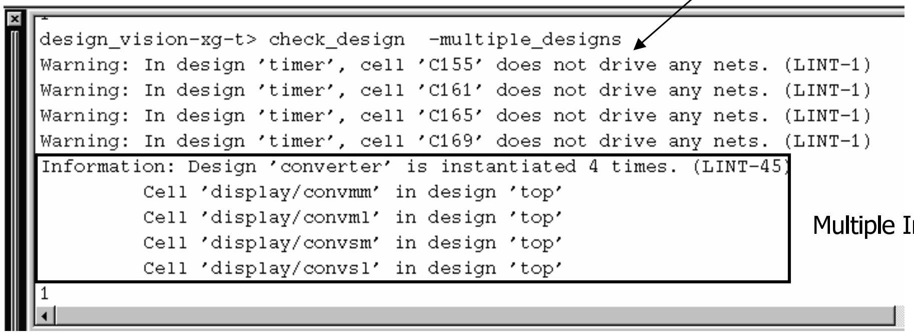
Setting DRC & Check Design

16.Setting area constraints and design rule constraints in#dc_shell-t command line
dc_shell-xg-t>#set_max_area 0
dc_shell-xg-t>#set_max_fanout 2 [all_inputs]
dc_shell-xg-t>#set_max_transition 0.3 [all_inputs]

17. Check design
dc_shell-xg-t>#check_design -multiple_designs
What's the warning message?________________

18.#Uniquify the design converter to fix the above warning
select#**top**
Hierarchy#->#Uniquify -> Hierarchy
OK

Cell C155, C161, C165, C169這些 Cell合成後就會被最佳化掉了 所以在此可以忽略掉!


```
design_vision-xg-t> check_design  -multiple_designs
Warning: In design 'timer', cell 'C155' does not drive any nets. (LINT-1)
Warning: In design 'timer', cell 'C161' does not drive any nets. (LINT-1)
Warning: In design 'timer', cell 'C165' does not drive any nets. (LINT-1)
Warning: In design 'timer', cell 'C169' does not drive any nets. (LINT-1)
Information: Design 'converter' is instantiated 4 times. (LINT-45)
        Cell 'display/convmm' in design 'top'
        Cell 'display/convml' in design 'top'
        Cell 'display/convsm' in design 'top'
        Cell 'display/convsl' in design 'top'
1
```

Multiple Instance問題

Report all Constraints Before Compiler

19. Generate the Port and Design reports to see whether the attributes have been# properly set
    Design#> Report Design,#
    Design#> Report Port (click on verbose),
    Design#> Report Clocks
    OK

20. Save design and setup file
    Save design
    Select#**top**
    File#> Save As
    Set File Name as#**top_before_compile.ddc**, File Format as#**ddc**
    Click on#**Save All Designs in Hierarchy**
    Save
    Save setup file
    File#> Save Info#> Design Setup
    top_setup.dc
    Save

    Examine the#top_setup.dc file in your#unix shell
    unix% more #top_setup.dc   (or   vi #top_setup.dc)

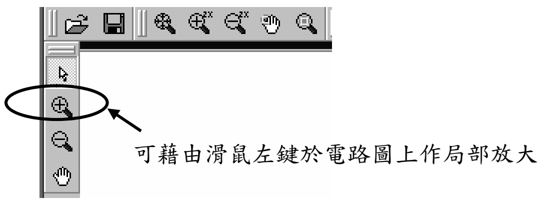
Compile Design

21. Compile design
    Select#**top**
    Design#> Compile Design
        set Map / Area Effort as#**high**
        OK
    (or#dc_shell-xg-t>#compile #bou   -map_effort high #area_effort high)

    Examine which designs were optimized? this is what known as "Hierarchical Compile"

22. Explore the schematic
    "**Create Design Schematic**" to see the result after synthesis.#

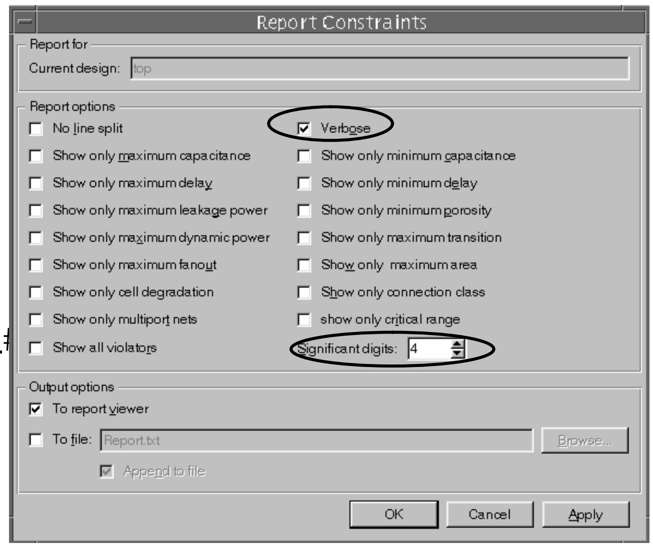
23. Work with some view commands or zoom tool
    View#> Zoom#> Zoom In
    View#> Zoom#> Zoom Out
    View#> Zoom#> Full View
    View#> Zoom#> Zoom to Selection



24. Select#**top**
    File#> Save As
    Set File Name as#**top_after_compile.ddc**, File Format as#**ddc**
    Click on#**Save All Designs in Hierarchy**
    Save

Report and Find Critical Path

25. Check the area, timing, constraints#
    - Select#**top**
    - **Design ➔ Report Area**
    - **Timing#➔ Report Timing**
    - **Design#➔ Report Constraints (click on verbose)**
    - Whether all constraints meet?________
    - What's the#cell area of this design?________#
    - Which is the critical path?________



Report Constraints

Report for
Current design: top

Report options
- [ ] No line split
- [✓] Verbose
- [ ] Show only maximum capacitance
- [ ] Show only minimum capacitance
- [ ] Show only maximum delay
- [ ] Show only minimum delay
- [ ] Show only maximum leakage power
- [ ] Show only minimum porosity
- [ ] Show only maximum dynamic power
- [ ] Show only maximum transition
- [ ] Show only maximum fanout
- [ ] Show only maximum area
- [ ] Show only cell degradation
- [ ] Show only connection class
- [ ] Show only multiport nets
- [ ] show only critical range
- [ ] Show all violators

Significant digits: 4

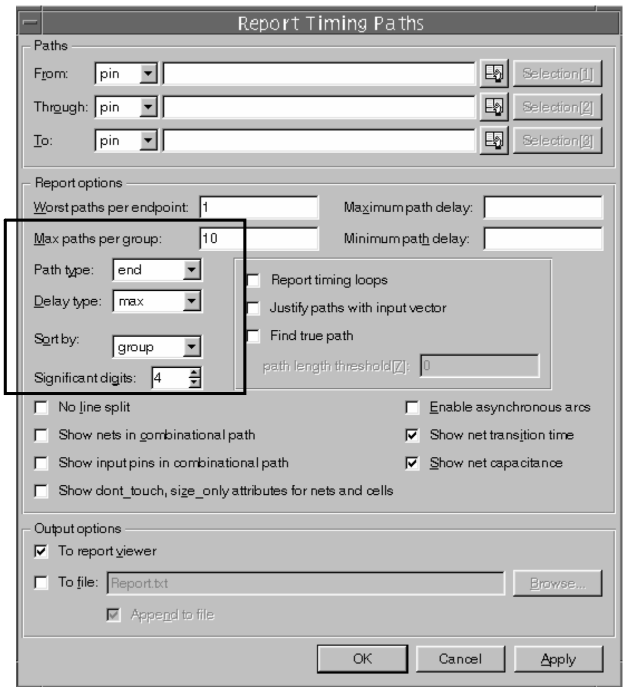
Output options
- [✓] To report viewer
- [ ] To file: Report.txt
- [✓] Append to file

26. Examine the critical path
    - Click the#**Create Design Schematic**#button, will show the schematic view of top
    - **View#➔ Highlight#➔ Critical Path**
    - Remove highlight
    - **View#➔ Highlight#➔ Clear All**

Report#– Multiple Timing Path

27. Generate another version of a timing# report
    - **Timing#➔ Report Timing**
    - Max paths per group: **10**
    - Path type:#**end**
    - **OK**
    - Examine if there are just one critical# path? ________________________

28. Go back top, generate report of# Reference, Hierarchy for top
    - **Design ➔ Report Design Hierarchy**
    - **Design ➔ Report Reference**



Report Timing Paths

Paths
From: pin
Through: pin
To: pin

Report options
Worst paths per endpoint: 1
Maximum path delay:
Max paths per group: 10
Minimum path delay:
Path type: end
Delay type: max
Sort by: group
Significant digits: 4
- [ ] Report timing loops
- [ ] Justify paths with input vector
- [ ] Find true path
path length threshold[%]: 0
- [ ] No line split
- [ ] Enable asynchronous arcs
- [ ] Show nets in combinational path
- [✓] Show net transition time
- [ ] Show input pins in combinational path
- [✓] Show net capacitance
- [ ] Show dont_touch, size_only attributes for nets and cells

Output options
- [✓] To report viewer
- [ ] To file: Report.txt
- [✓] Append to file

Dynamic Power Optimization

29. Before power optimization, you can see the timing & area & power first.
    cell area = ________ , #**timing**= ________ , #**power**= __________________#

30. Setting power constraints and gate-level power optimization
    dc_shell-xg-t>#set_max_total_power 0#uw
    dc_shell-xg-t> compile #-inc

31. After power optimization, you can see the timing & area & power.
    cell area = ________ , #**timing**= ________ , #**power**= __________________
    Dynamic Power Improve Capability(%)= ________ ,#
    Timing meet constraint? ________#

Save File#– gate-level#netlist

32. Save compiled design as db file and#verilog netlist
    File#-> Save As
    Set File Name as#**top_compile.ddc**, File Format as#**ddc**
    Click on#**Save All Designs in Hierarchy**
    Save

    File#-> Save As
    Set File Name as#**top_compile.v**, File Format as#**verilog**
    Click on#**Save All Designs in Hierarchy**
    Save

    File#-> Save Info#-> Design Timing
    SDF version:#**1.0**
    File name:#**top.sdf**

    Check if there is any warning message

33. Examine the#verilog file you just created
    In your#unix window, type#**more#top_compile.v**
    unix% more#top_compile.v

Lab 2-1 Answers / Solutions

```
17. Warning: Design 'converter' is instantiated 4 times. (LINT-45)
    Cell 'display/convmm' in design 'top'
    Cell 'display/convml' in design 'top'
    Cell 'display/convsm' in design 'top'
    Cell 'display/convsl' in design 'top'

21. 從Design最上層top開始一直到其最下層之所有block全部都會被合成與最佳化!

25. No (area constraint not meet)
    Cell Area: 1649 um2
    Critical Path: Start Point: test -> End Point: timer/sec_lsb_next_reg[0]/D

27. 最長的Path Delay(Critical Path)確實只有一條,但是從Report Timing不難發現,仍有相當多條的
    Path其Delay相當逼近Critical Path.

29. Cell area = 1649 um2, timing=3.8767ns(met constraint),
    power= dynamic: 133.0847 uw, leakage: 4.1396 uW

31. Cell area =1649 um2, timing=3.9675ns(met constraint),
    power= dynamic:120.5193 uW, leakage:2.9017 uW
    Dynamic Power Improve= 9.44%
    Timing meet constraint? Yes  (Power改善許多, Timing/Area仍不太受影響)
```

Lab2-2: Leakage Power Opt. by Multi-V_t (UMC90)

- In this lab, we will learn how to reduce the leakage power using the CIC providing UMC90 LVT/RVT/HVT standard cell.
- The design is the same as lab2-1

These labs take approximately 0.5 hour to complete

Multi-V_t Leakage Power Opt.

1. Change directory to synthesis run directory
   unix%#cd ~/SYNOPSYS/lab2/lab2-3/

2. View the .synopsys_dc.setup file, and then start Design Vision by command#
   unix% more .synopsys_dc.setup
   unix%#dv

關於**.synopsys_dc.setup** 的部分~
有空請同學自行比較跟**lab2-1**的設定有何不同處**!**
順便學習**UMC90 Multi-Vt**的設計方法**!**

3. Execute script file to Read Design & Set Constraints & Synthesis
   design_vision-xg-t> source #script.tcl

4. Before power optimization, you can see the timing & area & power first.
   cell area = ________ , #**timing**= ________ ,#**power**= _________

5. Setting power constraints and gate-level leakage power optimization
   dc_shell-xg-t>#set_max_total_power 0 #nw
   dc_shell-xg-t> compile #inc

6. After power optimization, you can see the timing & area & power.
   cell area = ________ , timing= ________ , power= ______________________
   Dynamic Power Improve (%)= ____________ ,
   Leagage Power Improve(%)= _____________ , Timing Meet? ________

Lab 2-2 Answers / Solutions

```
4.  Cell area =1645 um², timing=3.7174ns(meet timing constraint),
    Power= dynamic:165.3075#uw, leakage:19.9845#uw

6.  Cell area = 1657 um², timing=3.9970ns(meet timing constraint),
    Power= dynamic:107.0491#uw, leakage:1.7245 uW
    Dynamic Power Improve= 35.15%, Leakage Power Improve= 91.37%
    Meet Timing constraint? Yes (Leakage Power大幅度改善, Area/Timing仍然差不多!)
   (從本實驗中可以看出如果你有越多種Vt的Lib, 就盡量用, 然後交給DC去合成, 可以省下很多Power喔!)
```

Lab2-3: DC-Topographical (UMC90)

- In this lab, you must use the newer version such as DC. 2007.03 or DC 2007.12
- It is future tendency which a no wire load model design
- The design is same to lab2-1

很重要請務必要練習:

由於教室沒有DC 2007.03以後的版本, 請同學回家後自己練習

DC-topographical Flow

1. Change directory to synthesis run directory
   unix% cd ~/SYNOPSYS/lab2/lab2-3/

2. View the .synopsys_dc.setup and script file. What's the different of script.tcl between lab2-1 and lab2-3?
   ____

   unix% vi .synopsys_dc.setup
   unix% vi script.tcl

3. Enter the design vision topo mode
   unix% dv -topo

4. Execute script file to Create Milkway & Read Design & Set Constraints & Synthesis
   design_vision-topo> source script.tcl

   (注意: 第一次做本lab, 因爲compile_ultra會分析你所有的"db"file並建立"alib"目錄, 所以會比較久!)

5. After synthesizing, you can report the timing & area & power.
   cell area = ________ , timing= ________ , power= ________________

   What's the different of synthesizing result between lab2-1 and lab2-3?
   ____

Lab 2-3 Answers / Solutions#

2. 在.synopsys_dc.setup裡多了一行設定:
   set_tlu_plus_files -max_tluplus lib/tluplus/u90.tluplus#tech2itf_map lib/tluplus/u90.map

   在script.tcl裡多了一行設定:
   create_mw_lib COOK_MW #technology lib/umc_90nm_1p9m126_CIC.tf #open #
   -mw_reference_library "lib/FSD0A_A_GENERIC_CORE lib/FOD0A_B33_T33_GENERIC_IO"

   另外,#在合成的地方改成用**compile_ultra –no_auto**當作合成指令, 而不是**compile**.
   (-no_auto表示不要做auto_ungroup功能!)

5. Cell area = 1504 um^2, timing=3.36ns(meet timing constraint),
   Power= dynamic:67.65#uw, leakage:3.334 uW
   (本實驗做出來的數據表示,如果你是用Synopsys APR(ex: IC Compiler/Astro)軟體畫Layout,當Layout完成後,在Timing方面其效能也能達到約3.36ns之等效能!我們從lab2-1可知,使用不準的WLM來做設計,實在太過悲觀,Net Delay計算過大,這會照成你在做設計時,往往會誤以為自己做的結果太差無法達到預期效能因而重做或花時間再修改架構,而造成無謂的時間浪費!因此強烈建議同學以後都要改用DC-Topographical Flow來做設計!像我本人設計都已經改用此流程設計了!)

   (另外,使用DC-T Flow也會使Power分析出來的結果較為正確喔!WLM所估的Power完全沒參考依據!)

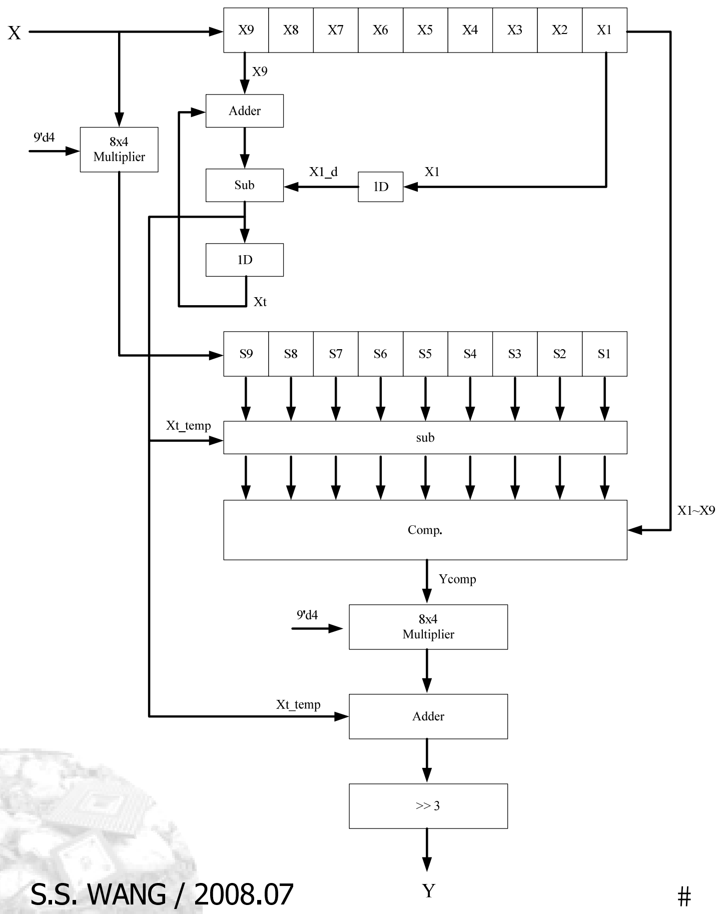
(DC 2007.12製作的解答)

Labs: CHIP Level Design Using tsmc18

- Lab 3-1: ICC 2005 (Cell-Based Contest) – Top Level Synthesis
- Lab 3-2: ICC 2005 (Cell-Based Contest) – CHIP Level Synthesis
- Lab 3-3: ICC 2005 (Cell-Based Contest) – DC-T CHIP Synthesis

These labs take approximately 1.5 hour to complete

Calculation System Block Diagram



其他說明事項(Any other information you want to specify:(如設計特點 ...)

1、關於 Calculation System Block Diagram 如圖一所示。

2、從圖一可得知，本設計使用了三個技巧:

2.1 採用乘法器取代除法器之複雜運算，一來可改善除法器面積之浪費，同時在 Critical Path 上也可避掉一個除法器的時間，大幅減少 Combination Delay 時間。

2.2 改變輸出Y_j之分子部分，大幅減少題目中之式(4)加法器之使用量，並減少運算時間。

2.3 將輸出Y_j之分母部分，用” >>3” ，取代除法器(/8)之動作。

3、簡易動作流程如下:

3.1 求$Xavg_j$、$Xappr_j$時，原先是九筆數相加後再相除，在本設計中則是將每筆數都先乘九，再與這九筆數之總和做比較，找出最短距離之數字就是$Xappr_j$。

3.2 將輸出Y_j之分子部分，原式為$[(X_1+Xappr)+(X_2+Xappr)+ \quad ... \quad +(X_9+Xappr)]$改成$(X_1+X_2+...+X_9)+9Xappr=X_t+9Xappr$，以減少加法器之使用量及加快題目中式(4)分子部分的運算速度。

3.3 將步驟 3.2 加完後之結果使用” >>3” ，取代除法器(/8)之動作，其輸出即為所求。

Lab 3-1: Top-level Synthesis

1. Change directory to lab3-1
   unix% cd ~/SYNOPSYS/lab3/lab3-1

2. View the lab3-1 design directory
   unix% cd design
   unix% vi CS.v
   unix% vi CHIP.v

3. Change directory to syn_dc and view all *.tcl files
   unix% cd ~/SYNOPSYS/lab3/lab3-1/syn_dc

```
unix% vi 00_run.tcl        (run all script file)
unix% vi 01_import.tcl     (import all design & Core-Level Constraint)
unix% vi 02_compile.tcl    (compile design & DFT in XG mode Constraint)
unix% vi 03_output.tcl     (save file script file)
```

Run Script file to finish Synthesize

4. Change directory to synthesis run directory
   unix% cd run

若覺得機器太慢者, 建議可直接讀取**cs_syn.ddc**擋案, 然後繼續第六步驟!
方法: **File -> Read ..** 點選**cs_syn.ddc**, **open**
(回去可改用**Linux**機器**Run,** 約**28**秒可執行完**)**

5. Execute all script files to finish this lab

```
unix% dv               -f ../00_run.tcl  | tee run.log
      dc_shell-xg-t -f ../00_run.tcl  | tee run.log
```

6. Check the area, timing
   Design ➔ Report Area
   What's the cell area of this design?________
   Timing ➔ Report Timing
   What's the timing slack of CHIP-Level (current_design CHIP)?____________
   What's the timing slack of CORE-Level(current_design CS)?____________

7. Have these file that "cs_syn.ddc" & "cs_syn.vg" & "chip.sdf" & "chip_syn.spf" existed? If they have existed, you can exit Design Compiler.

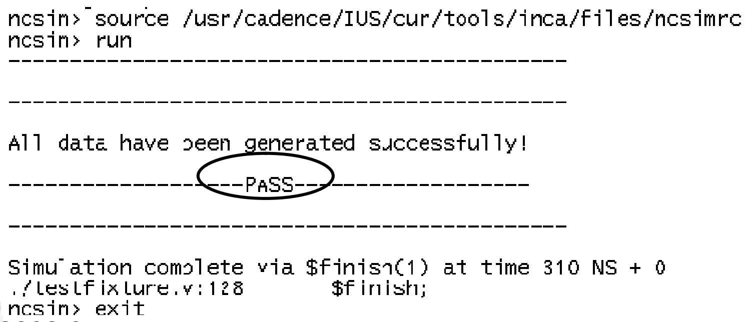
Timing Simulation

8. Change directory to Pre-Sim directory
   unix% cd ~/SYNOPSYS/lab3/lab3-1/tbench/presim

9. Link synthetic output result: cs_syn.vg & chip.sdf for pre-layout simulation
   unix% ln -s ../../syn_dc/run/cs_syn.vg .
   unix% ln -s ../../syn_dc/run/chip.sdf .

10. View test bench file and add sdf to this file (在此已改好,所以不用另外加入!)
    initial $sdf_annotate(chip.sdf, chip);

11. Begin to simulate synthetic result. Is this function PASS? (有興趣者可用**nWave**開**.vcd**來看波形)
    unix% ncverilog testfixture.v cs_syn.vg -v ../tsmc18.v -v ../tpz973g.v +access+r



```
ncsim> source /usr/cadence/IUS/cur/tools/inca/files/ncsimrc
ncsim> run
---------------------------------------------

---------------------------------------------

All data have been generated successfully!

-------------------PASS-------------------

---------------------------------------------

Simulation complete via $finish(1) at time 310 NS + 0
./testfixture.v:128         $finish;
ncsim> exit
```

Lab 3-1 Answers / Solutions

```
6.  Cell Area= 273011.953750 um2
    CHIP Timing= 19.9781 ns
    CORE Timing= 19.9994 ns
    (若CORE & CHIP Timing差異很多, 表示CHIP Level Boundary Constraint需再加強.
     這樣才能將此CHIP_dc.tcl檔案交給APR Tool接續作Layout,當作APR時要給的Constraint檔案)
```

Lab 3-2: CHIP-level Synthesis#

1. Change directory to lab3-2
   unix%#cd ~/SYNOPSYS/lab3/lab3-2

2. First step, the chip-level synthesis must be writing the CHIP module#
   unix%#cd ~/SYNOPSYS/lab3/lab3-2/syn_dc
   unix% vi#CHIP.v
   What's the input/output pad cells used in this example? ____________
   What's the input/output pad cell name in this example? ____________

(本範例中**CHIP module** 已寫好,在此無需再寫)

3. Invoke the Design Vision XG mode
   unix%#cd run
   unix%#dv

4. View the 01_import.tcl script file,#
   what's different from the lab3-1's 01_import.tcl? ________________________
   ____

5. View the 02_compile.tcl script file,#
   what's different from the lab3-1's 02_compile.tcl? ________________________
   ____

Run Script file to finish Synthesize

6. View the 03_output.tcl script file,#
   what's different from the lab3-1's 03_output.tcl? ________________________
   ____

7. Execute all script files to finish this lab
   dc_shell-xg-t> source #../00_run.tcl #

若覺得機器太慢者,#建議可直接讀取**cs_syn.ddc** 擋案,#然後繼續第八步驟!
方法:#**File#-> Read#..** 點選**cs_syn.ddc,#open**
(回去可改用**Linux**機器**Run,#**約**42**秒可執行完)

8. After chip-level#synthezing, check the area, timing
   Design ➔ Report Area
   What's the#cell area of this design?________#
   Timing#➔ Report Timing #
   What's the timing slack of CHIP-Level (current_design CHIP)?____________
   Is the synthesizing result the same with lab 3-1? ________________________

9. Have these file that#"**cs_syn.ddc**" &#"**cs_syn.vg**" & #"**chip.sdf**" &
   "**chip_syn.spf**" existed? If they have existed, you can exit Design Compiler.

Timing Simulation

10. Change directory to Pre-Sim directory

    unix%#cd ~/SYNOPSYS/lab3/lab3-2/tbench/presim

11. Link synthetic output result:#cs_syn.vg &#chip.sdf for pre-layout simulation

    unix%#ln -s ../../syn_dc/run/cs_syn.vg .

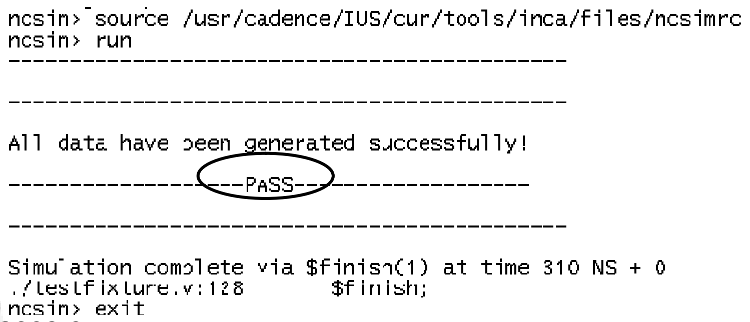
    unix%#ln -s ../../syn_dc/run/chip.sdf .

12. View test bench file and add#sdf to this file #(在此已改好,#所以不用另外加入!)

    initial $sdf_annotate(chip.sdf, chip);

13. Begin to simulate synthetic result. Is this function PASS?#(有興趣者可用**nWave**開**.vcd**來看波形)

    unix%#vcs –R #testfixture.v cs_syn.vg -v ../tsmc18.v #v ../tpz973g.v



```
ncsim> source /usr/cadence/IUS/cur/tools/inca/files/ncsimrc
ncsim> run
---------------------------------------------

---------------------------------------------

All data have been generated successfully!

------------------PASS------------------

---------------------------------------------

Simulation complete via $finish(1) at time 310 NS + 0
./testfixture.v:128        $finish;
ncsim> exit
```

Lab 3-2 Answers / Solutions

```
2.  Input Pad Cell: PDIDGZ ;Output Pad Cell: PDO08CDG
    Input Cell Name:  ipad_clk, ipad_reset, ipad_si, ipad_se, ipad_X7~ipad_X0
    Output Cell Name: opad_Y9~opad_Y0, opad_so

4.  lab3-1 是切換到top-level (CS),準備給予Top-level constraints (CORE_dc.tcl)
    lab3-2 是切換到chip-level (CHIP),準備要給予CHIP-level constraints (CHIP_dc.tcl)

5.  第一點:
    lab3-1 是切換到top-level (CS),準備從top-level作Synthesis
    lab3-2 是切換到chip-level(CHIP),準備從chip-level作Synthesis
    第二點:
    lab3-2比lab3-1多了下列幾行:
     set_dont_touch [get_cells ipad*]
     set_dont_touch [get_cells opad*]
    (因為本範例的CHIP.v所有Input/output pad Cell Name分別是用ipad/opad開頭,
     因此不用下太多行的set_dont_touch指令! 此乃技巧處!)

6.  lab3-1 是做Top-level Synthesis,所以作Chip-level存檔還要再給個Chip-level Constraint
    (CHIP_dc.tcl),否則對Chip-level而言沒有任何的Constraint, sdf存檔時就會有問題!

    lab3-2是做CHIP-level Synthesis,合成前就已給定CHIP_dc.tcl了!在此無須再給!


8.  Cell Area= 269419.062500 um²
    CHIP Timing= 19.9930 ns
    Yes, Area小一些, Timing一樣是meet的! 但唯一不同處就是Chip-level比較方便又簡單,可以不用
    寫Top-level constraint (such as CORE_dc.tcl).
```

Lab3-3: DC-Topographical (TSMC18)

- In this lab, you must use the newer version such as DC. 2007.03 or DC 2007.12
- It is future tendency which a no wire load model design
- The design is same to lab3-2

很重要請務必要練習:

由於教室沒有DC 2007.03以後的版本, 請同學回家後自己練習

DC-topographical Flow

1. Change directory to synthesis run directory
   unix% cd ~/SYNOPSYS/lab3/lab3-3/syn_dc

2. View the .synopsys_dc.setup and script file. What's the different of script file between lab3-2 and lab3-3?

   ____

   unix% vi run/.synopsys_dc.setup
   unix% vi 01_import.tcl
   unix% vi 02_compile.tcl

3. Enter the design vision topo mode
   unix% cd run
   unix% dv -topo -f 00_run.tcl | tee run.log

   (注意: 第一次做本lab, 因爲compile_ultra會分析你所有的"db"file並建立"alib"目錄, 所以會比較久!)

4. After synthesizing, you can report the timing & area & power.
   cell area = ________ , timing= ________ , power= ________________

   What's the different of synthesizing result between lab3-2 and lab3-3?

   ____

Lab 3-3 Answers / Solutions#

2. 在.synopsys_dc.setup裡多了一行設定:
   set_tlu_plus_files #max_tluplus lib/tluplus/t18.tluplus #tech2itf_map lib/tluplus/t18.map

   在01_import.tcll裡多了一行設定:
   create_mw_lib CS_MW #technology lib/tsmc18_CIC.tf #open #
   -mw_reference_library "lib/tsmc18_fram/ lib/tpz973g/"

   在02_compile.tcll裡合成的地方,#要將compile#scan成用**compile_ultra -scan #no_auto**
   (-no_auto 就是不要做auto_ungroup功能)

4. Cell area = 253039.880521 um^2, timing=19.9851ns(meet timing constraint),
   Power= dynamic:20.97 mw, leakage:1.344 uW
   (本實驗做出來的數據表示,如果你是用Synopsys APR(ex: IC Compiler/Astro)軟體畫Layout,當Layout完成後,在Timing方面其效能也能達到約19.9851ns之等效能!強烈建議同學以後都要改用DC-Topographical Flow來做設計!像我本人設計都已經改用此流程設計了!)

   (另外,使用DC-T Flow也會使Power分析出來的結果較為正確喔!WLM所估的Power完全沒參考依據!)

(DC 2007.12製作的解答)

Labs: Retiming and Pipeline Design

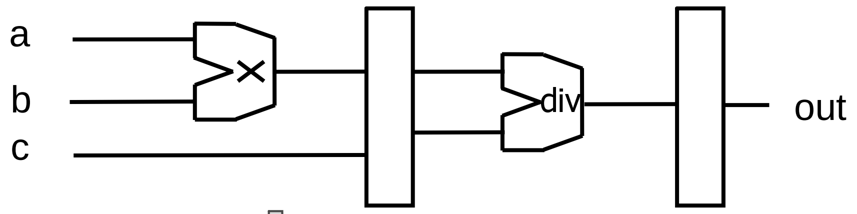
- Lab 4-1: Register Retiming
- Lab 4-2: Pipeline Design

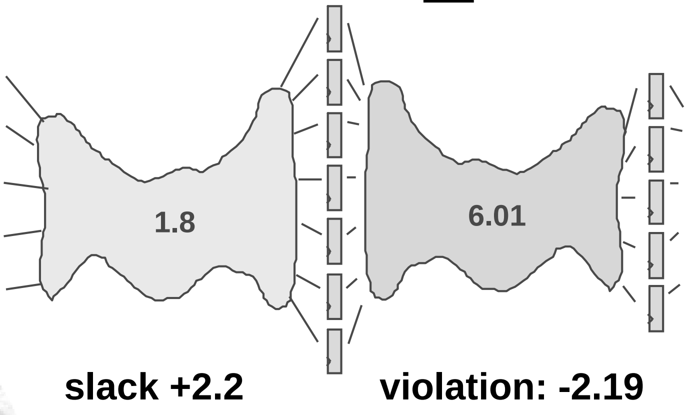
These labs take approximately 1 hour to complete

Lab 4-1: Retiming Registers

- How can we speed up the arithmetic? by retiming registers

```
out <= (a*b)/c;
```





assume

clock period = 4 ns
Clk->q delay = 0.2 ns
setup time = 0.1 ns

Getting Start

1.# Enter the directory **SYNOPSYS/lab4**, then list the directory:#**ls -al**
unix%#cd ~/SYNOPSYS/lab4
unix%#ls –al

2. Check the#**90nm UMC/Faraday** library contents of#**.synopsys_dc.setup**.
unix% more .synopsys_dc.setup

3. Invoke the Design Vision XG mode
unix%#dv

4. Check the search path, link library, target library and symbol library. We will use# **fsd0a_a_generic_core_wc.db** &#**fsd0a_a_generic_core_bc.db** as link# library and target library,#**generic.sdb** as symbol library.
File#-> Setup If it is ok,#**Cancel**

5. Read files
File#-> Read
Format: Auto
Click on#**muldiv.v**
Open

Report Timing Before Retiming

6. Setting the design constraints
Examine the#script.tcl file in your#unix shell
unix% more #script.tcl

What's the period of clock in#script.tcl file? _______

File#->#Execute_Script
Click on #**script.tcl**
Open

由於SUN機器太慢(要跑約330分鐘),#所以該步驟在此可先不做,# 直接讀取**muldiv_syn_no_retiming.ddc**檔案即可!
File#-> Read#.. 點選**muldiv_syn_no_retiming.ddc, open**
(有興趣的同學可以回去用Linux機器跑!約7.2分鐘)

7. Explore the schematic
"**Create Design Schematic**" to see the result after synthesizing.

8. Check the area, timing
Select#**top**
Design ➔ Report Area
Timing#➔ Report Timing
What's the#cell area of this design?________#
What's the timing of this design?________
Does the design meet timing constraint? __________ , slack= _________

Register Retiming

9. Solve too few module port connections problem
   Select **top**
   dc_shell-xg-t> remove_unconnected_ports -blast_buses [get_cells * hier]

10. Save file
    File -> Save As
    Set File Name as **muldiv_before_retiming.ddc**, File Format as **ddc**
    Click on **Save All Designs in Hierarchy**
    Save

11. Retiming Register (DC-Ultra Instruction)
    dc_shell-xg-t> optimize_registers

12. Explore the schematic
    "**Create Design Schematic**" to see the result after retiming registers.

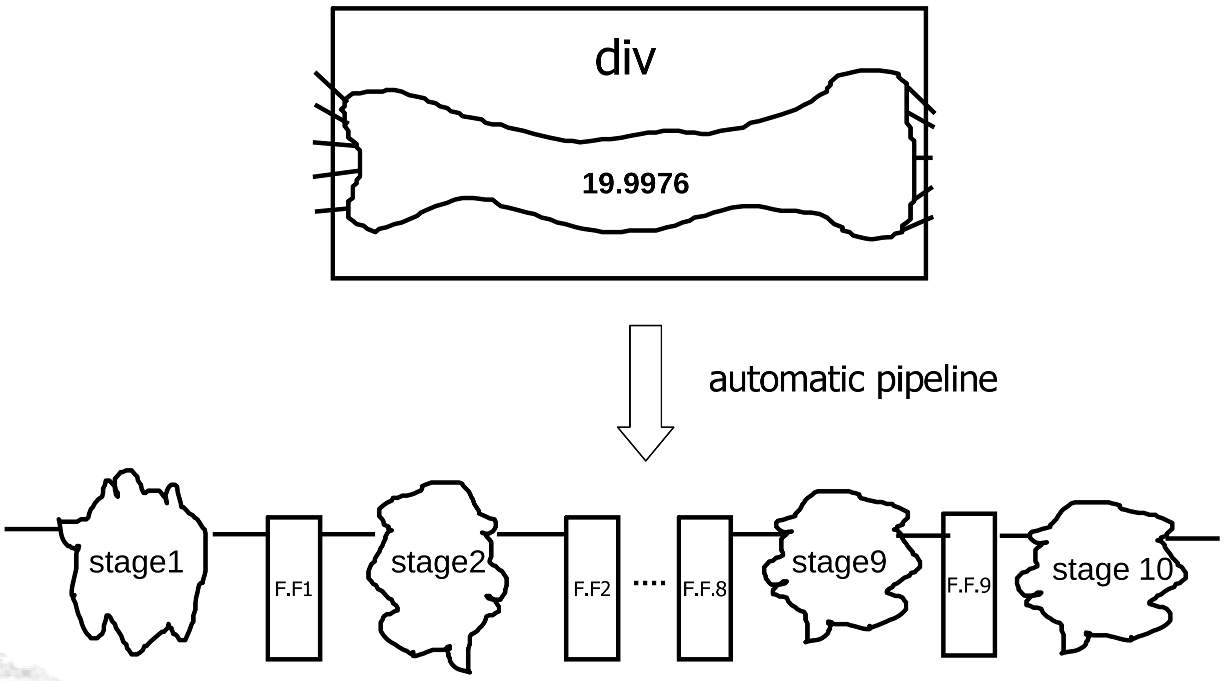
13. Check the area, timing
    Design ➔ Report Area
    Timing ➔ Report Timing
    What's the cell area of this design?________
    What's the timing of this design?________
    Does the design meet timing constraint? __________ , slack= _________

Lab 4-1 Answers / Solutions

```
6.  4 ns
8.  cell area= 27032 um2
    timing= 6.3886 ns
    No
    slack= -2.3886 ns
13. cell area= 29797 um2
    timing= 3.9999 ns
    Yes, meet timing constraint (retiming可以輕鬆且快速立即改善Timing, 面積沒差多少!)
    slack= 0.0001 ns
```

Lab 4-2:#Pipeline Design

- How can we speed up the combination circuit?#pipeline design#



Getting Start

1.# Remove all design
 File#-> Remove All Designs

2. View the#div.v file in your#unix shell
 unix% more #div.v
 Which port is redundancy in the design? ________ and ________

3. Read files
 File#-> Read
 Format: Auto
 Click on#**div.v**
 Open

4. Setting Timing Constraint
 dc_shell-xg-t>#**set_max_delay 20 #-from [all_inputs] #-to [all_outputs]**

5. Compile design
 dc_shell-xg-t>#**compile #-ungroup_all**

Pipeline Design and Analysis

6. Explore the schematic
   "**Create Design Schematic**" to see the result after synthesis.

7. Check the timing
   Select **div**
   Timing 🡺 Report Timing
   What's the timing of this design?________

8. Pipeline Design (DC-Ultra Instruction) in dc_shell-t>
   pipeline_design -period 2 -stages 10 -clock_port_name clk -async_reset rst

9. Explore the schematic
   "**Create Design Schematic**" to see the result after pipelining design.
   How is the ports of "clk" and "rst" now? ___________________________________

10. Check the timing
    Select **div**
    Timing 🡺 Report Timing
    What's the timing of this design?________ ,
    Does the design meet timing constraint? ________ , slack= ________ , latency= ________

Lab 4-2 Answers / Solutions

```
2.  clk and rst
7.  timing= 19.9976ns
9.  clk & rst port跟Pipeline Register已相連
10. timing= 1.9994ns
    Yes, meet timing constraint (Timing部份,改善了大約10倍的速度)
    slack= 0.0006 ns
    latency= 9  (因爲切了10 stages Pipeline, 會多九排的F.F.)
```