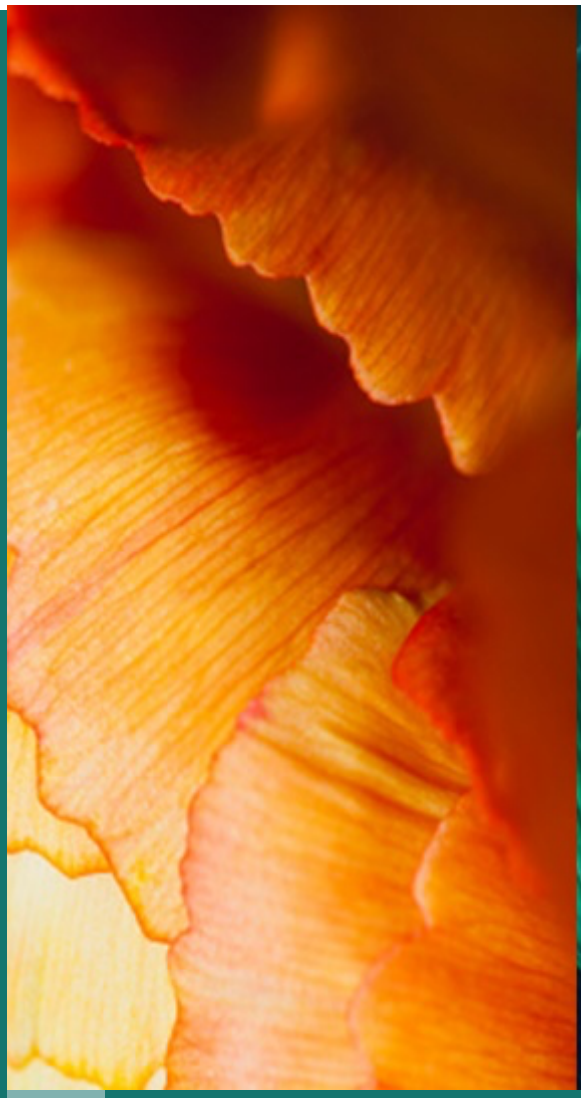


Design Compiler SOP

編輯日期: 2022/12/30

編寫者: Louis

Setup

```
remove_design -designs                                  #clear
set disable_multicore_resource_checks true
set_host_options -max_cores 16                          #多核心加速合成
set design  CHIP                                        #.v 檔名稱
set syn_design  CHIP_syn                                #合成後要存檔的名稱
```

Read file (Method1)

```
sh mkdir model                                  #先建資料夾存中間檔(.mr .pvl .syn)
define_design_lib model -path ./model           #設定 model 路徑
analyze -library model -format verilog ./${design}.v -autoread
                                #分析 verilog 可否合成, 有 include .v 要加 autoread
elaborate ${design} -architecture verilog -library model    #讀入.v 檔
link                                            #檢查是否有子 module 沒有 include

*若 design 有從 top module 更改子 module 之 parameter 請使用方法一
```

Read file (Method2)

```
read_file  -format  verilog  ./${design}.v          #讀入.v 檔
current_design  ${design}                           #切換至 top module
```

Setting before synthesis

```
*此部分視個人需求自行設定即可*
uniquify                                #若呼叫同個子 module 多次,幫其重新命名
source ${design}.sdc                                    # sdc 自行參考 ic 競賽
set_max_area        0                                   #不管將面積壓到最小
set_max_fanout      10    [all_inputs]
set_max_transition  0.3   [all_inputs]
set_dynamic_optimization true                           #針對動態功耗優化
set_leakage_optimization true           #Multi-Vt 改善靜態功耗(for TN40)
set_fix_multiple_port_nets -all -buffer_constants [get_designs *]
        #design 中若有 output=常數 or input 等問題,補 buffer,避免合成後
         netlist 裡出現 assign
set enable_keep_signal true                 #合成後不要拆解特定 signal
set hdlin_keep_signal_name user             #請在.v 中不要拆解的 signal 後
                                加上 // synopsys keep_signal_name "訊號名字"
```

Synthesis

```
*自行選擇要開啟的選項*
compile -map_effort high \              #優化程度(越高 CPU Time 越久)
-area_effort high \                     #Area 優化優先
-power_effort high \                    #Power 優化優先
-gate_clock \                           #使用 clock gating 技術
-scan \                                 #加入 scan-chain 並優化(for DFT)
-boundary_optimization \                #刪除沒用的 CKT 並做 global 最佳化
-inc    #小幅度優化(不做 logic 優化只做 gate-level 優化),第二次合成在使用
-auto_ungroup area                      #將 cell 數少之 module 拆掉做最佳化
-ungroup_all                           #全 module 拆掉做最佳化

compile_ultra                   #更低面積(40%↓),更快速度(20%↓)
-no_autoungroup                 #不要把 module 全拆做最佳化
```

Switching Activity Information Format (SAIF)

```
vcd2saif -input ${design}.vcd -output ${design}.saif
                                #跑完 tb 後在 terminal 下此指令,將 vcd 轉 saif
read_saif -input ${design}.saif -inst    tb 名稱/instance name
                                #回到 dc_shell,引入 saif 檔
*若無引入 saif 檔,dc 的 report_power 是不準的,因為 toggle rate 預設 50%
*更進階 power 分析可跑 PTPX
```

Report

```
report_design                    #看設定環境(lib、PVT、wireload model)
report_area  -hier                #看面積
report_timing   -delay min        #看 setup time violation
\ -delay max                      #看 hold time violation
\ -path end -max_paths 5          #看前 5 名 critical path
report_power                      #注意要先引入 saif 檔
report_net_fanout -high_fanout    #看 high fanout(>1000) signal
\ -threshold 50                   #看 fanout >50) signal
report_qor
```

Output

```
define_name_rules name_rule -case_insensitive     #避免大小寫問題
change_names -hierarchy -rules name_rule          #將特殊符號換掉
change_names -hierarchy -rule verilog

write -format ddc -hierarchy -output ${syn_design}.ddc     # 輸出 ddc
write -hierarchy -format verilog -output ${syn_design}.v   #輸出.v
write_sdf -version 2.1 -context verilog -load_delay net ${syn_design}.sdf
                                                           #輸出 sdf
```

```
write_sdc    ${syn_design}.sdc                                    #輸出 sdc
write_scan_def -output ${syn_design}.scandef
                                                  #輸出 scan-chain 資訊,APR 要用
```

Clock Gating Summary

Before Synthesis

```
set_clock_gating_style -control_point before          #在 icg 前加入控制點
\ -setup 0.3            #預留 Setup margin，沒下 gate-level 模擬可能會 fail
\ -minimum_bitwidth 3                                  #DFF 超過幾 bit 換 icg
\ -control_signal scan_enable
                                        #只有 shift in & out 時為 1, capture 時為 0
\-control_signal  test_mode                                    #always 為 1
\-observation_point true  #加入觀測點(xor tree), 使用 test_mode 要加
\ -num_stages 2                                    #Multi-stage clock gating
\ -max_fanout 128                  #避免 fanout 太大導致 clk delay 太久
set_clock_gating_check -setup 0.3
 #檢查 icg 是否符合 setup time，因為 dc 預設只檢查 DFF 有沒有符合 setup time，參閱下圖
```

Synthesis

```
compile -gate_clock
report_clock_gating -gating_elements                    #看 gating rate
report_clock_gating_checks      #檢查對 icg 的 setup check 是否有設到
```

Before DFT

```
set_dft_clock_gating_pin -pin_name TE [get_cells -hier clk_gate*]
                                          #讓 DC 辨別 icg 的 TE pin
rewire_clock_gating -proximity                    #讓 icg 都接最近的 DFF
\ -balance_fanout                      #讓每個 icg 接的 DFF 數量平均一點
```

If gate-level simulation is not pass, check command in .sdc

set_clock_latency 0.0 [all_clocks]

set_ideal_network [get_ports clk]

Set delay from ICG/ENCLK to DFF/CK 0 after synthesis

report_net_fanout -threshold 30

Net	Fanout	Attributes	Capacitance	Driver
clk	52	dr, d, I	0.00	clk
pe/net343	271	dr, d	3.04	pe/clk_gate_flower_inf_r_reg/main_gate/Y
pe/net348	271	dr, d	3.04	pe/clk_gate_other_inf_r_reg/main_gate/Y
pe/net358	64	dr, d	0.72	pe/clk_gate_man_mask_r_reg/main_gate/Y
pe/net363	64	dr, d	0.72	pe/clk_gate_overlap_mask_r_reg/main_gate/Y

1

set_ideal_network [get_pins all_clock_gates -out_pins]

or change size of ICG after synthesis

set temp [get_cells * -hier -filter {clock_gating_style == latch}]

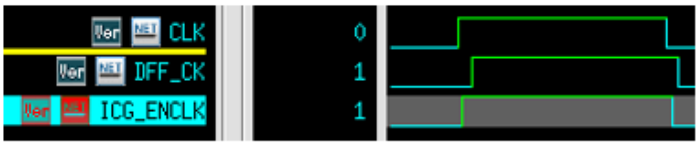
#找出所有 icg 的 latch,存在變數 temp 中

size_cell $temp TLATNTSCAX20 #將 icg 的 latch 換成 driving 更大的版本

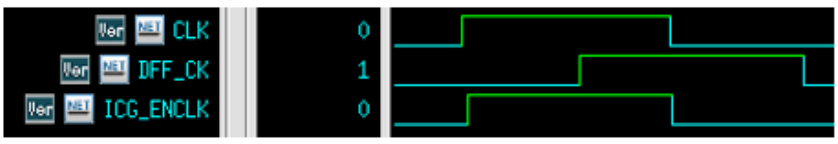
#cell 名稱自行查詢製程檔

Ideal_network 設置前

Low_fanout_ICG

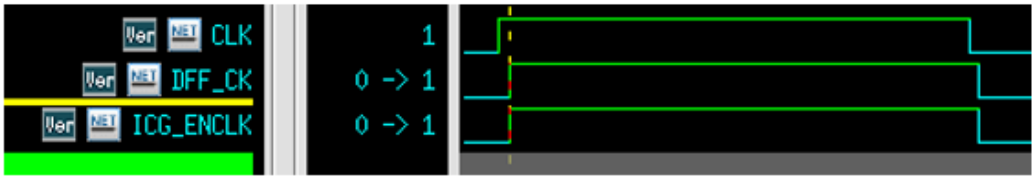


High_fanout_ICG



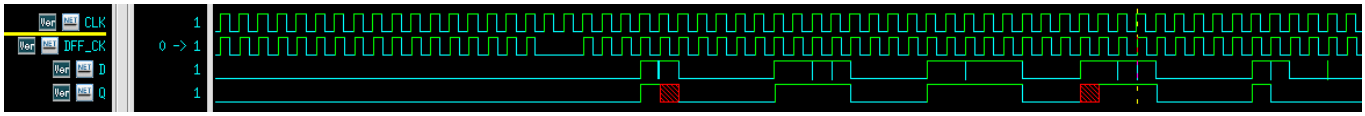
設置後

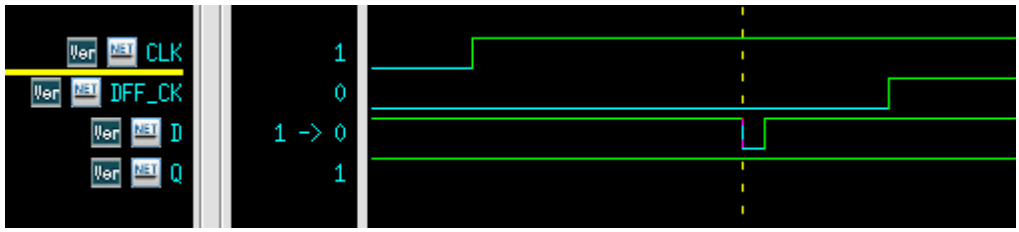
High_fanout_ICG



set_clock_gating_style -setup 0.3 設置前

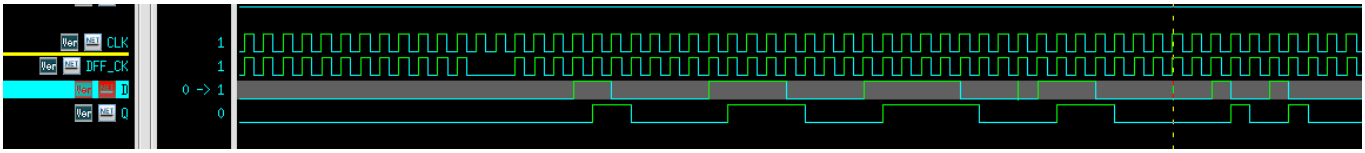
因為 high fanout icg 會有 delay,導致多個 glicth 雖滿足 clk 的 setup time，但不滿足 DFF_CK 的 setup time





set_clock_gating_style -setup 0.3 設置後

加了後會把 glitch 修掉，便滿足其 setup time



```
Point                                                          Incr       Path
-----------------------------------------------------------------------------
clock clk (rise edge)                                          0.00       0.00
clock network delay (ideal)                                    0.00       0.00
sgde_ctrl/curr_state_reg_2_/CK (DFFRX4)                        0.00       0.00 r
sgde_ctrl/curr_state_reg_2_/QN (DFFRX4)                        0.64       0.64 r
sgde_ctrl/U69/Y (CLKINVX2)                                     0.63       1.27 f
sgde_ctrl/U277/Y (NAND2X1)                                     0.69       1.96 r
sgde_ctrl/U334/Y (NOR2X1)                                      0.30       2.27 f
sgde_ctrl/U333/Y (CLKINVX1)                                    0.24       2.51 r
sgde_ctrl/U331/Y (AOI21X1)                                     0.17       2.68 f
sgde_ctrl/clk_gate_LO_cnt_reg/EN (SNPS_CLOCK_GATE_HIGH_sgde_ctrl_3)
                                                               0.00       2.68 f
sgde_ctrl/clk_gate_LO_cnt_reg/latch/D (TLATNX1)                0.00       2.68 f
data arrival time                                                         2.68
```

Other

```
help *關鍵字*                                                    #查相關指令
printvar *關鍵字*                                               # 查相關的變數設定
man  指令                                          #關於此指令或變數設定的完整使用說明
sizeof_collection [get_pins */Q -hier]                    #統計使用幾個 Reg
set_false_path -from PIN_NAME -to PIN_NAME
#設定某條 path 不做 STA 分析
set HIGH_FANOUT_NET [all_high_fanout -threshold 1000 -nets]
set_ideal_network -no_propagate [remove_from_collection
$HIGH_FANOUT_NET {clk rst*}]
#忽略 high fanout net 的 delay
```

Multi-Vt Summary

```
*請在.dc_setup 檔中的 set target_library 中加入 lvt & rvt 的.db 製程檔
set_attribute [get_libs HVT_LIBNAME]  -type string \
default_threshold_voltage_group hvt
set_attribute [get_libs LVT_LIBNAME]   -type string \
default_threshold_voltage_group lvt
#LIBNAME 自行查詢
report_threshold_voltage_group                  #看 lvt hvt 在電路中的比例


set_leakage_optimization true
```

Auto Synthesis using For Loop

```
for {set i 4} {$i<=16} {incr i 1} {
set cycle [expr ($i+4)*0.5]                    # 自行設定要合成的 cycle
remove_design -designs
read_file -format verilog {/home/louis/soc/ef/CHIP.v}

將 sdc 內容貼上於此


compile_ultra
report_area -hierarchy > data/area[$i].txt      #自行先建個 data 的資料夾
uplevel #0 { report_timing -path full -delay max -nworst 1 -max_paths 1
-significant_digits 2 -sort_by group } > data/time[$i].txt
```