Synthesis with Synopsys D.C.

Tool Command Language

Format

```
"command_name" + "variable" + "argument_0" + "argument_1" + ...
```

Return

```
all variables are in string format
```

Example

```
set love 520
echo "love Iclab"
```

Get variable's value

```
$variable_name
```

```
set b 37; # b = "37"
set a b;  # a = "b"
set c $b; # c = "37"
```

Nested commands

```
set b 8;           # b = "8"
set a [expr $b+2]; # a = "10"
```

Linux commands in dc_shell

```
sh + commands
```

```
sh mkdir XXX
```

Comment

Remember to add ";" between commands and comment

```
set a 3; # a = "3"
```

get_ and all_ commands

```
get_inputs, get_outputs, get_ports ...
all_inputs, all_outputs, ...
```

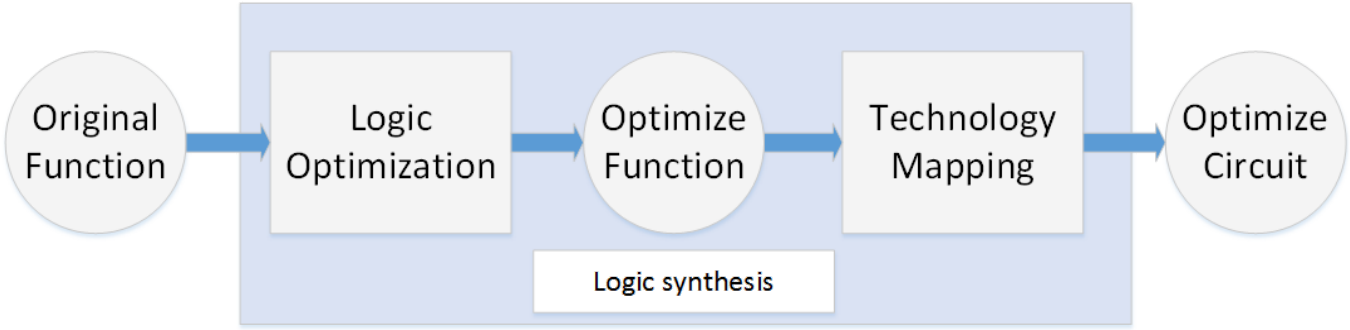
List the designs, ports, reference, cells, clocks, I/O, pins ...

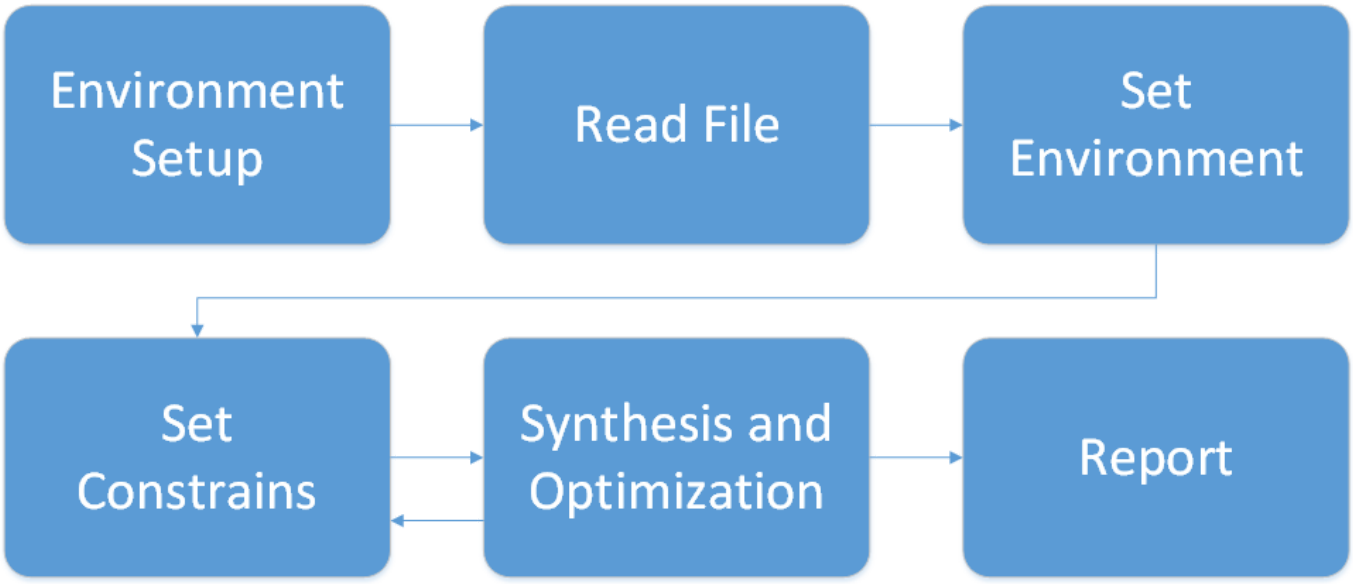
```
get_inputs {B* D*}
```

Help command

```
help "command name";       # 內建說明文件
man "command name";        # 輔助使用說明
printvar "variable name"; # 輸出變數內容
echo $"variable name";     # 輸出變數內容
```

Synthesis Flow





.synopsys_dc.setup

Used for initializing design parameters and variables. Command in this file will be executed when DC is invoked.
We will set :

1. Libray path which will be used during synthesis
2. Basic log, environment setup

```
set search_path "xxxxx"                 # 合成時要用到的所有 library 所放置的檔案路徑
set target_library "yyyyy"              # 合成時會用到的，所要對應到的 library
set link_library "zzzzz"                # 合成前要參考的 library (對應可以使用的
design ware)

set symbol_library "?????.sdb"          # 合成後用來表示結果外觀的 library
set synthetic_library "??????.sldb"     # 合成時要把 RTL map 到電路的 library

set suppress_errors [concat $suppress_errors "UID-401"] # 當對於port設定driving
cell的時候
                                                        # 會對於該port 的timing 估
計造成影響
                                                        # 因此DC預設會對於這樣的指令
顯示錯誤。
                                                        # 在這裡將設定將這個錯誤忽
略，讓合成流程更順暢。

# Environment Settings
set command_log_file "./command.log";           # 設定log檔儲存的地方
set view_command_log_file "./view_command.log"; # 設定log檔儲存的地方

set hdlin_translate_off_skip_text "TRUE"; # 避免對於 translate off block 中的 code
做檢查
set edifout_netlist_only "TRUE";          # 輸出netlist時，以EDIF格式輸出，該格式用
於不同EDA tool之間的交流規範
set verilogout_no_tri true;               # 把輸出的netlist中，tri-state的腳位改用
```

```
wire宣告

set sh_enable_line_editing true;          # 開啟自動補字
set sh_line_editing_mode emacs;           # 設定文字編輯器
history keep 100;                         # 要保存的歷史指令數
alias h history;                          # 指定h為history
```

synthesis.tcl

```
# 重要的參數在外面設定，讓腳本可以盡量重複使用
set TOPLEVEL "top_module_name"
set TEST_CYCLE 3.2
source -echo -verbose 0_readfile.tcl # 把全部的腳本依照功能分階段管理
source -echo -verbose 1_setting.tcl  # 我們會在後面依序完成
source -echo -verbose 2_compile.tcl
source -echo -verbose 3_report.tcl
exit
```

0_readfile.tcl

創建資料夾來管理所有在 synthesis 的檔案 (log, report, netlist ...)

```
set TOP_DIR $TOPLEVEL
set RPT_DIR report
set NET_DIR netlist

sh rm -rf ./$TOP_DIR
sh rm -rf ./$RPT_DIR
sh rm -rf ./$NET_DIR
sh mkdir ./$TOP_DIR
sh mkdir ./$RPT_DIR
sh mkdir ./$NET_DIR
```

創建一個 library 來記錄這個 design

```
define_design_lib $TOPLEVEL -path ./$TOPLEVEL
```

Read and elaborate design

```
# Read Design File (add your files here)
set HDL_DIR "../source"
analyze -library $TOPLEVEL -format verilog "???????" # put all your HDL here
elaborate $TOPLEVEL -architecture verilog -library $TOPLEVEL
```

Solve multiple instance，uniquify 的過程會將 module 對應到新的名字，因此需要指定命名規則

```
# Solve Multiple Instance
set uniquify_naming_style "%s_mydesign_%d"
uniquify
```

Link

```
# 指定要處理的 design 下一步要設定整個電路的 constrain，所以要在 top 做
current_design $TOPLEVEL
# 連結 design 跟 library
link
```

1_setting.tcl

Set environment

```
# Setting Design and I/O Environment
# 設定操作環境 SS corner 0.95V 125C
set_operating_conditions -library saed32hvt_ss0p95v125c ss0p95v125c
```

Set wire load model

```
# 設定線的 wire load model
set auto_wire_load_selection area_reselect
# 用 enclosed 的方式來決定 sub blocks 間的 wire
set_wire_load_mode enclosed
# 以面積做為選擇 group 的判斷
set_wire_load_selection_group predcaps
```

Specify clock

```
# Setting Timing Constraints
# 創建名為 clk 的 clock，週期是 $TEST_CYCLE 並連接到所有 clk 的 ports 上
create_clock -name clk -period $TEST_CYCLE  [get_ports clk]
# ideal_network 忽略訊號 driving 能力問題
set_ideal_network       [get_ports clk]
# 在這條路徑上不因時間考量而加入 buffer
set_dont_touch_network  [all_clocks]
```

Set Interface delay

```
# 設定 I/O 兩端所連接電路的 delay
# I/O delay should depend on the real environment
# 針對除了 clk 外的所有 primary I/O
set_input_delay  [expr $TEST_CYCLE/2.0] -clock clk [remove_from_collection
[all_inputs] [get_ports clk]]
set_output_delay [expr $TEST_CYCLE/2.0] -clock clk [all_outputs]
```

Other constrains

```
# Setting DRC Constraint
# Defensive setting: smallest fanout_load 0.041 and WLM max fanout # 20 =>
0.041*20 = 0.82
# 依據一些假設設定最大 fanout load (WLM 最多有 20 個 fanout)
# max_transition and max_capacitance are given in the cell library
set_max_fanout 0.82 $TOPLEVEL

# Area Constraint
# 面積盡量小
set_max_area        0
```

2_compile.tcl

```
#this cell's will cause ncverilog to hang
set_dont_use [format "%s%s" saed32hvt_ss0p95v125c {/SDFFNASRX1*}]
```

```
# 在 STA 時，如果某些訊號為 constant 就直接以常數來分析，不把它當作變數
set case_analysis_with_logic_constants true
# 避免直接連結的 net 中出現 assign 的描述 (backend 會有問題)，所以以 buffer 取代
set_fix_multiple_port_nets -feedthroughs -outputs -constants -buffer_constants
```

Check design

```
# Checks for possible timing problems in the current design
check_design > ./$RPT_DIR/check_design.log
# Checks the current design for consistency
check_timing > ./$RPT_DIR/check_timing.log
```

```
# 對於 gated clock，設定控制邏輯閘的最大 fanout
set_clock_gating_style -max_fanout 10
```

Synthesis all design

```
# gate_clock 省電
# exact_map 不將 logic 簡化到 sequential 中
# no_autoungroup 避免為了化簡而分解 module，all hierarchies are preserved
# no_boundary_optimization 避免為了化簡而使某些 module 的 output pin inversion
# no_seq_output_inversion
# 這些大部分是為了 LEC 可以驗證合成結果
compile_ultra -exact_map -no_autoungroup -no_seq_output_inversion -
no_boundary_optimization

# In the incremental mode, the tool does not run the mapping or implementation
selection stages
# compile_ultra -incremental -exact_map -no_autoungroup -no_seq_output_inversion -
no_boundary_optimization
```

```
# remove dummy ports
remove_unconnected_ports [get_cells -hierarchical *]
# decompose buses and remove dummy ports
remove_unconnected_ports [get_cells -hierarchical *] -blast_buses
```

3_report.tcl

- Set naming rules
- Export netlist, delay information, design constrains ...
- Export synthesis results
- Detail commands refer to 3_report.tcl