ASIC FULL FLOW

1) **Preparation**
   - Tạo cây thư mục (đặt tên bất kỳ) bằng lệnh: mkdir -p counter/rtl chẳng hạn
2) **RTL Design & Verification**
   - Truy cập thư mục tổng vừa tạo
   - Tạo đường dẫn “rtl”: mkdir rtl
   - Truy cập đường dẫn đó và add 2 files: rtl design + testbench
   - Code mẫu:
     - rtl

counter.v

```v
`timescale 1ns / 1ps
////////////////////////////////////////////////////////////////////////////////
module counter (
    input   wire            clk,
    input   wire            rst_n,
    input   wire            i_enable_w,
    input   wire            i_updown_w,     // 1: up, 0: down
    output  reg     [7:0]   o_count_r       // 0 -> 255
);
    reg     [7:0]   count_next_r;

    wire    [7:0]   count_up_1_w;
    assign          count_up_1_w = o_count_r + 8'd1;

    wire    [7:0]   count_down_1_w;
    assign          count_down_1_w = o_count_r - 8'd1;

    //Combinational logic to determine next count value
    always @(*) begin
        if (i_enable_w) begin
            if (i_updown_w) begin
                count_next_r = count_up_1_w;
            end else begin
                count_next_r = count_down_1_w;
            end
        end else begin
            count_next_r = o_count_r;
        end
    end

    //Sequential logic to update count value on clock edge
    always @(posedge clk or negedge rst_n) begin
        if (!rst_n) begin
            o_count_r <= 8'd0;
        end else begin
            o_count_r <= count_next_r;
        end
    end
endmodule
```

     - testbench

tb_counter.v

```v
`timescale 1ns/1ps

module tb_counter;

    reg clk;
    reg rst_n;
    reg i_enable_w;
    reg i_updown_w;

    // DUT output
    wire [7:0] o_count_r;

    // Instantiate DUT
    counter dut (
        .clk        (clk),
        .rst_n      (rst_n),
        .i_enable_w (i_enable_w),
        .i_updown_w (i_updown_w),
        .o_count_r  (o_count_r)
    );

    // Clock generation: 10ns period (100 MHz)
    initial begin
        clk = 0;
        forever #5 clk = ~clk;
    end

    // Stimulus
    initial begin

        // Initial state
        rst_n      = 0;
        i_enable_w = 0;
        i_updown_w = 1;

        // Hold reset
        #20;
        rst_n = 1;

        #10 i_enable_w = 1;
        i_updown_w = 1;   // UP
        #100;

        i_updown_w = 0;   // DOWN
        #100;

        // Disable counter
        i_enable_w = 0;
        #50;

        rst_n = 0;
        #20;
        rst_n = 1;

        i_enable_w = 1;
        i_updown_w = 1;
        #100;

        $finish;
    end

endmodule
```

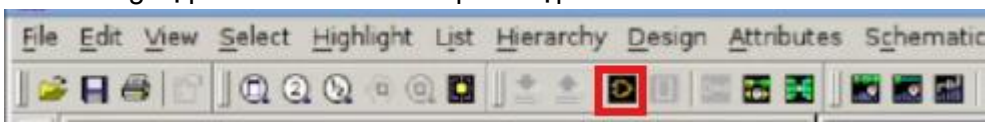
- Về thư mục tổng, tạo đường dẫn “sim”: mkdir sim
- Truy cập đường dẫn “sim” vừa tạo, tạo đường dẫn con “pre_syn”
- Truy cập đường dẫn “pre_syn”
- Dùng lệnh ‘vcs -debug +v2k ../../rtl/*.v để check syntax 2 files verilog trong rtl
- Sau đó bật DVE bằng lệnh ‘./simv -gui’ để simulate xem waveform
- Khi vào gui, chuột phải cửa sổ nhỏ bên trái có tên “tb_...(tùy theo tên testbench của ae)” > Add to wave > New waveview – Nhấn F5 trên bàn phím > Xem waveform xong > Tắt DVE
- Dùng pwd để thư mục đang đứng hiện tại
- Dùng cd ../../ để về thư mục cha counter
- Tạo đường dẫn “syn_check”: mkdir syn_check
- Truy cập syn_check

- Chạy lệnh ‘leda’ để bật Synopsys Leda – đây là tool check syntax nâng cao, cũng như xem code có synthesizable, ASIC friendly hay không. Ví dụ: inferred latch, sensitivity list có list đủ chưa và còn nâng cao ác hơn nữa ae
- Sau khi vào leda, cửa sổ project sẽ mở lên
- Chọn New project > OK
- Để đường dẫn cho Project name là “/home/ltk/counter/syn_check/counter.pro” > Next
- Tại cửa số HDL, trong Verilog chọn Version 2001 > Next > Next lần 2
- Tại cửa sổ Source Files, xuống ô Files chọn đường dẫn rtl chứa file rtl design > Next > Finish
- Sau khi bấm Finish, Leda sẽ bắt đầu chạy và out ra cửa sổ Specify Design Information, kéo chân input clk vào clk tích cực cạnh lên thì để rising, rst_n của anh để mức thấp thì để Low level > OK
- Exit qua Synthesis

3) Synthesis

- Về thư mục gốc counter, tạo đường dẫn “syn” bằng mkdir syn
- Truy cập syn
- Để mở Design Compiler, sử dụng lệnh ‘dc_shell -gui’
- Design Compiler mở, vào File > Setup > Add các thư viện tương ứng dưới đây
- Search path:
    - /home/ltk/ASIC_LIB/Lib/syn_lib/syn_lib
- Link library:
    - /home/ltk/ASIC_LIB/Lib/syn_lib/syn_lib/saed90nm_max.db
    - /home/ltk/ASIC_LIB/Lib/syn_lib/syn_lib/saed90nm_min.db
    - /home/ltk/ASIC_LIB/Lib/syn_lib/syn_lib/saed90nm_typ.db
      /home/eda/synopsys/syn_vD-2010.03-SP4/libraries/syn/dw_foundation.sldb
- Target library:
    - /home/ltk/ASIC_LIB/Lib/syn_lib/syn_lib/saed90nm_max.db
    - /home/ltk/ASIC_LIB/Lib/syn_lib/syn_lib/saed90nm_min.db
    - /home/ltk/ASIC_LIB/Lib/syn_lib/syn_lib/saed90nm_typ.db
- Symbol library:
    - /home/eda/synopsys/syn_vD-2010.03-SP4/libraries/syn/generic.sdb
- Synthetic library:
    - /home/eda/synopsys/syn_vD-2010.03-SP4/libraries/syn/dw_foundation.sldb
- Sau đó OK
- Dùng command ‘set_svf counter.svf’: để ghi lại mọi tối ưu trong quá trình synthesis vô file svf để dùng cho LEC – Formality Checking sau này
- File > Read.. > trỏ tới file RTL design trong folder rtl > OK
- File > Analyze > Add > file RTL design trong folder rtl > Format: Verilog HDL > Tick Create Library nếu kh exist > OK
- File > Elaborate > Design: chọn file top module nếu thiết kế có nhiều hierarchy > Tick Reanalyze… > OK
- Design > Check Design > OK (nhớ theo dõi console log)

- Bấm schematic Button trên thanh công cụ để xem mạch – lúc này mạch chưa chạy tối ưu tổng hợp nếu nhìn có vẻ rất phức tạp



- Bây giờ khi đã thấy được mạch bằng schematic > Chuột phải vô chân clock của mạch > Attributes trên tool > Specify Clock.. > Period 20ns để test (nếu slack dương mạnh thì giảm period để mạch chạy được Fmax cao nhất có thể) > Rising 0 Falling 10 > Tick Don't touch network + Tick Fix hold > OK – Này là chúng ta khai báo tần số đầu vào cho mạch á
- Tiếp theo, chọn tất cả chân input (không phải chân clock) cùng lúc ví dụ: reset, enable, up_down chẳng hạn > Attributes > Operating Environment > Input Delay > Max rise 2 Min rise 1 > Tick add delay > OK
- Chọn chân output > Attributes > Operating Environment > Output Delay > Max rise 2 Min rise 1 > Tich add delay > OK
- Vào Attributes > Operating Environment > Wire Load > Kéo lên đầu chọn cái đầu tiên > OK
- Vào Attributes > Operating Environment > Operating Conditions > Library: chọn typ Condition TYPICAL > OK
- Vào Attributes > Optimization Constraints > Design Constraints > Max area: 1000 > OK
- Design > Compile Design > Để setting Default > OK (Tuy nhiên anh không khuyến khích mấy đứa sử dụng Compile thường này mà hãy xài thẳng Compile Ultra và bỏ qua thằng này)
- Sau đó mấy đứa lại xem schematic lúc này các cổng đã được tối ưu
- Chạy thằng Compile Ultra: Design > Compile Ultra > OK (có thể tìm hiểu thêm về setting của nó)
- Check timing report: Timing > Report Timing Path > kiểm tra Path type: full Delay type: max (setup checking report) hoặc để min (hold checking report) > OK
- Mấy đứa có thể check report về constraints về resource, power bằng cách vào Design > Report …..
- Sau khi xong hết sẽ xuất ra file : File > Save as > nhớ check là đã trong thư mục syn – lưu lại tên file là "netlist.v" – đây là netlist mà synthesis xuất ra cho tụi mình để xài cho các stage tiếp theo – tương tự cho counter.ddc đây là file sẽ lưu netlist, constraints, attributes mà mình đã làm
- Dùng lệnh 'write_sdf counter.sdf' để lưu lại các delay cơ bản
- Dùng lệnh 'write_sdc counter.sdc' để lưu các timing và non timing constraint của mạch sau synthesis

4) Formality Checking – Logic Equipvalent Checking

- Make sure chúng ta đang về thư mục counter
- Tạo đường dẫn "formal" bằng mkdir formal
- cd formal
- Dùng lệnh fm_shell -gui để bật giao diện tool check formal
- Ở 0.Guidance, chọn Guidance > Chọn đường dẫn về thư mục syn – chọn file svf > OK > Load Files
- Qua 1. Reference, 1. Read Design Files – đây là mình sẽ load golden netlist tức là netlist từ rtl: Chọn Verilog > Về đường dẫn rtl chọn file counter.v > OK > Load Files

- Qua mục 3. Set Top design của 1. Reference: Chọn counter > Set Top
- Qua mục tổng 2. Implementation – đây là nơi mình sẽ load file netlist sau synthesis vào: Chọn Verilog > Về đường dẫn syn chọn file netlist.v > OK > Load Files
- Qua mục 2. Read DB: Chọn DBs > vào đường dẫn sau /home/ltk/ASIC_LIB/Lib/syn_lib/synlib – Ctrl + A sau đó load tất cả files DB trong đó vào > Load Files
- Qua mục 3. Set top design của 2. Implementation: chọn file counter_1 để Set Top
- Qua mục tổng 4. Match: Run Matching > 0 Unmatching Compare Points thì lụm tiền
- Qua mục tổng 5. Verify: Run Verify: check console thấy passing hết total thì lụm tiền

5) Post Synthesis – STA

- Ở terminal, về thư mục counter, mkdir timing
- cd timing
- chạy pt_shell để cho tool chạy bằng tcl command
- Trong terminal: chạy lần lượt các lệnh sau
    - set lib_path "/home/ltk/ASIC_LIB/Lib/syn_lib/syn_lib";
    - set ADDITIONAL_SEARCH_PATH "$lib_path";
    - set TARGET_LIBRARY_FILES "saed90nm_typ.db";
    - set_app_var search_path "$search_path $ADDITIONAL_SEARCH_PATH";
    - set_app_var target_library $TARGET_LIBRARY_FILES;
    - set_app_var link_library "* $target_library";
    - read_ddc /home/ltk/counter/syn/counter.ddc
    - read_sdc /home/ltk/counter/syn/counter.sdc
- Sau đó dùng lệnh 'start_gui' để mở giao diện tool ở dạng cửa sổ
- Vào Timing trên toolbar > Report Timing > OK (check report)

6) Post Synthesis – Simulation

- Truy cập thư mục sim của counter
- mkdir post_syn > cd post_syn
- Copy file netlist, sdf từ thư mục syn vào post_syn bằng lệnh:
    - cp -r ../../syn/netlist.v ./
    - cp -r ../../syn/counter.sdf ./
- Copy 90nm model vào post_syn bằng lệnh:
    - cp -r ../../../../ASIC_LIB/Lib/syn_lib/model/saed90nm.v ./
- Copy file testbench từ rtl vào post_syn:
    - cp -r ../../rtl/tb_counter.v ./
- Dùng lệnh 'gedit netlist.v &' để add thêm `timescale 1ns/1ps rồi save lại. Chú ý tên top module counter trong file này: counter_1
- 'gedit tb_counter.v &' để chỉnh tên instance thành counter_1. Lưu ý chỉ chỉnh ở bước này

```
// Instantiate DUT
counter_1 dut (
    .clk        (clk),
    .rst_n      (rst_n),
    .i_enable_w (i_enable_w),
    .i_updown_w (i_updown_w),
    .o_count_r  (o_count_r)
);
```

- Dùng lệnh ‘vlogan -debug +v2k *.v’ để phân tích kiểm tra cú pháp của file verilog trong post_syn
- Dùng lệnh ‘vcs -debug tb_counter -sdf typ:counter_1: counter.sdf -l comp.log’ để chuẩn bị môi trường, kết nối các module cho việc mô phỏng
- Dùng lệnh ‘./simv -gui’ để bật tool DVE: Khi vào gui, chuột phải cửa sổ nhỏ bên trái có tên “tb_...(tùy theo tên testbench của ae)” > Add to wave > New waveview – Nhấn F5 trên bàn phím > Xem waveform xong > Tắt DVE

7) Physical Design – Layout

- Về terminal, trong thư mục counter, mkdir layout > cd layout
- mkdir lib
- ‘icc_shell -gui’ để bật IC Compiler

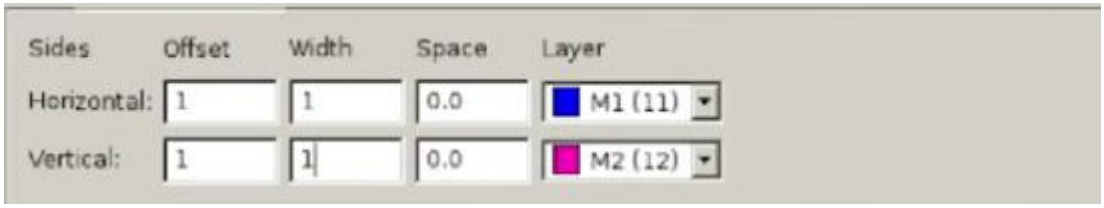
a) Data setup

- Vào File trên tool bar > Setup > Application Setup:
- Search path:
    - /home/ltk/ASIC_LIB/Lib/syn_lib/syn_lib
- Link library:
    - /home/ltk/ASIC_LIB/Lib/syn_lib/syn_lib/saed90nm_max.db
    - /home/ltk/ASIC_LIB/Lib/syn_lib/syn_lib/saed90nm_min.db
    - /home/ltk/ASIC_LIB/Lib/syn_lib/syn_lib/saed90nm_typ.db /home/eda/synopsys/syn_vD-2010.03-SP4/libraries/syn/dw_foundation.sldb
- Target library:
    - /home/ltk/ASIC_LIB/Lib/syn_lib/syn_lib/saed90nm_max.db
    - /home/ltk/ASIC_LIB/Lib/syn_lib/syn_lib/saed90nm_min.db
    - /home/ltk/ASIC_LIB/Lib/syn_lib/syn_lib/saed90nm_typ.db
- Symbol library:
    - /home/eda/synopsys/syn_vD-2010.03-SP4/libraries/syn/generic.sdb
- Synthetic library:
    - /home/eda/synopsys/syn_vD-2010.03-SP4/libraries/syn/dw_foundation.sldb
- OK
- File > Create Library > Ở new library path chọn thư mục lib trong layout vừa tạo ở trên > new library name: my_lib
- Technology file: /home/ltk/ASIC_LIB/Lib/process/astro/tech/astroTechFile.tf
- Tick Input reference libraries > Add :
    - /home/ltk/ASIC_LIB/Lib/process/astro/fram/saed90nm_fr > OK
- Tick Open library > OK
- File > Import > Read DDC > Add > Chọn counter.ddc từ thư mục syn > OK
- File > Import > Read SDC > Input file name: Chọn counter.sdc từ thư mục syn > OK
- Windows > New Layout Window
- Trong Layout Window, File > Set TLU+:
    - Max TLU+ file: /home/ltk/ASIC_LIB/Lib/process/star_rcxt/tluplus/saed90nm_1p9m_1t_Cmax.tluplus
    - Min TLU+ file: /home/ltk/ASIC_LIB/Lib/process/star_rcxt/tluplus/saed90nm_1p9m_1t_Cmin.tluplus
    - Layer name mapping file between technology library and ITF file: /home/ltk/ASIC_LIB/Lib/process/astro/tech/tech2itf.map

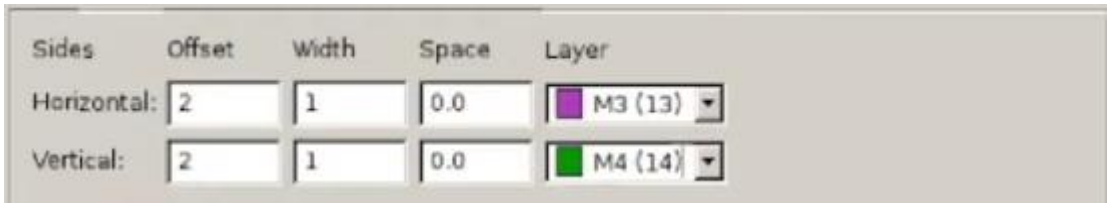
- OK
- Vô console của tool: ‘check_library’
- ‘check_tlu_plus_files’ trong console
- ‘check_timing’ trong console
- ‘report_timing_requirements’ trong console
- ‘report_disable_timing’ trong console
- ‘report_case_analysis' trong console
- ‘report_clock’ trong console
- ‘report_clock -skew’ trong console
- ‘remove_ideal_network [get_ports]' trong console
- ‘save_mw_cel -as data_setup’ trong console

b) Floorplan

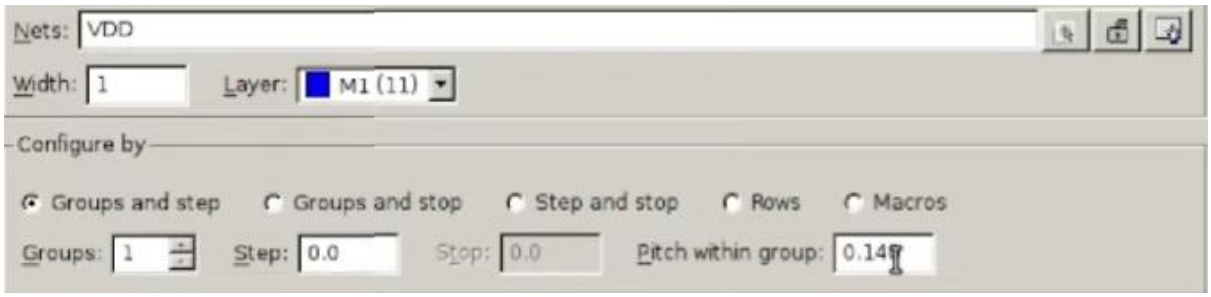
- Vào layout window, Floorplan > Initialize Floorplan > Core to Left, Right, Top Bottom set là 5 hết > OK
- Preroute > Derive PG Connection > Tick Manual connection > Power net: VDD, Ground net: VSS, Power pin: VDD, Ground pin: VSS > Create port: Tick All > OK
- Preroute > Create Rings > Net: VDD > Như hình dưới > OK



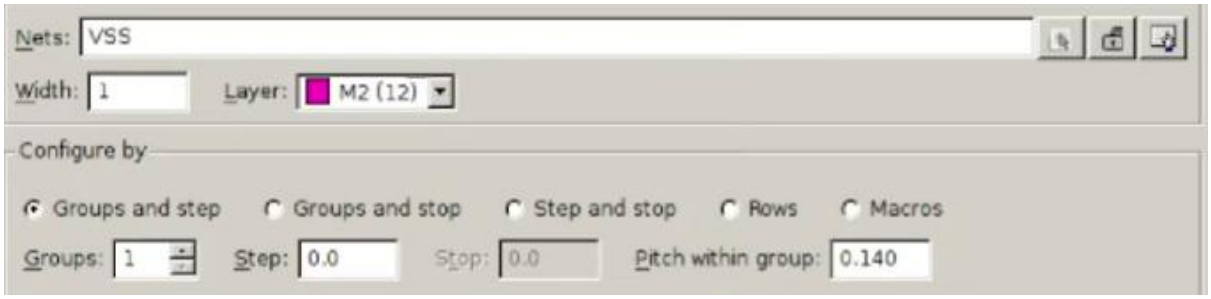
- Preroute > Create Rings > Net: VSS > Như hình dưới > OK



- Preroute > Create Power Straps > Nets: VDD > Như hình dưới 0.140 > OK



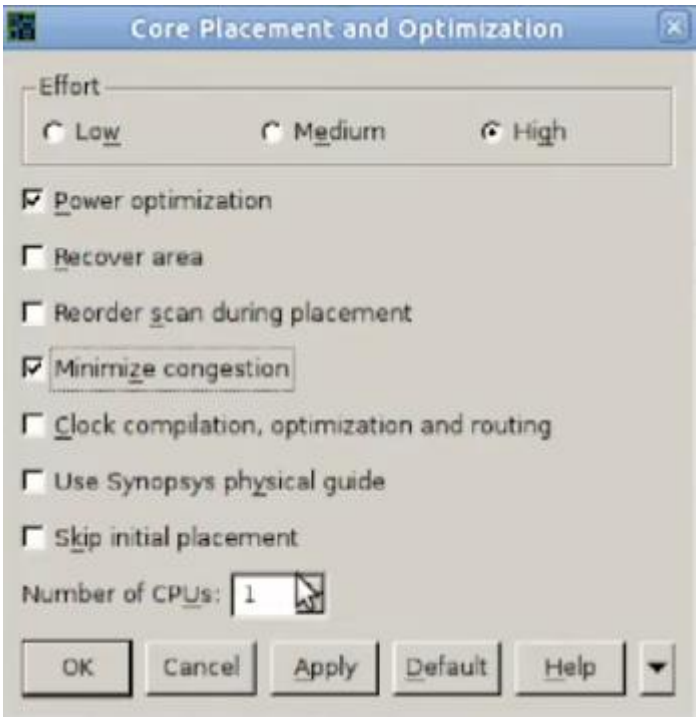
- Preroute > Create Power Straps > Nets: VSS > Như hình dưới 0.140 > OK



- Để save ‘save_mw_cel -as floorplan’

c) Placement

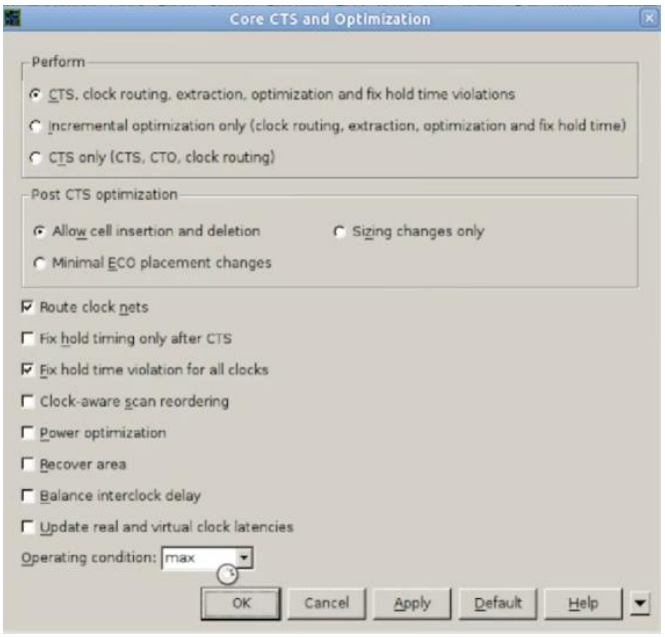
- Placement > Core Placementand Optimization > Tick Effort: High, Power Optimization, Minimize Congestion > OK



- Lúc này có thể standard cell đã được move vào bên trong core area
- Chạy globalroute để check overflow bằng lệnh:
    - report_congestion -grc_based -by_layer -routing_stage global
- Save design tại stage ‘ save_mw_cel -as placement’

d) Clock Tree Synthesis – CTS (route metal đường clock)

- Clock > Core CTS and Optimization > Tick fix hold time violation for all clocks > OK



- Có thể quan sát trên layout lúc này đường clock signal đã được route metal
- ‘save_mw_cel -as cts’

e) Routing (Route metal tất cả đường signal)

- Route > Core Routing and Optimization > Effort: High Tick Perform power optimization > OK
- ‘save_mw_cel -as routing’
- Route > Extract RC > OK mục đích để có thông tin chính xác của metal net delay cho giai đoạn STA Signoff
- Trong ICC console: icc_shell> ‘write_verilog icc.v’
- ‘write -format ddc -hierarchy -modified -output icc.dcc’
- ‘write_sdc icc.sdc’
- ‘write_sdf icc.sdf’