2015 University/College IC Design Contest

Cell-Based IC Design Category for Graduate Level

Image Sorting Engine

1.問題描述

請完成一 Image Sorting Engine(**後文以 ISE 表示**)的電路設計。此電路可將 32 張固定影像尺寸之二維(2D)彩色影像訊號，作顏色的分類成紅色、綠色、藍色三大類別，並將其影像按照暗紅色…排列到亮紅色，暗綠色…排列到亮綠色，暗藍色…排列到亮藍色，排列後將影像所屬之索引 (index)值，依序輸出。有關 ISE 詳細規格將描述於後。

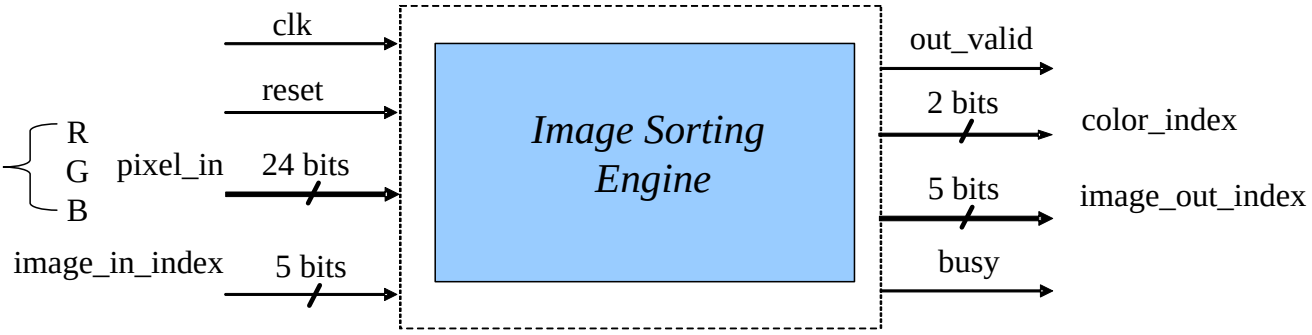
本電路各輸入輸出信號的功能說明，請參考表一。每個參賽隊伍必須根據下一節所給的設計規格及附錄 A 中的測試樣本完成設計驗證。

本次 IC 設計競賽比賽時間為上午 08:30 到下午 20:30。當 IC 設計競賽結束後，CIC 會根據第三節中的評分標準進行評分。為了評分作業的方便，各參賽隊伍應參考附錄 E 中所列的要求，附上評分所需要的檔案。

本題目之測試樣本置於 ***/usr/cad/icc2015/bgc/icc2015cb.tar*** ，請執行以下指令取得測試樣本：

tar xvf /usr/cad/icc2015/bgc/icc2015cb.tar

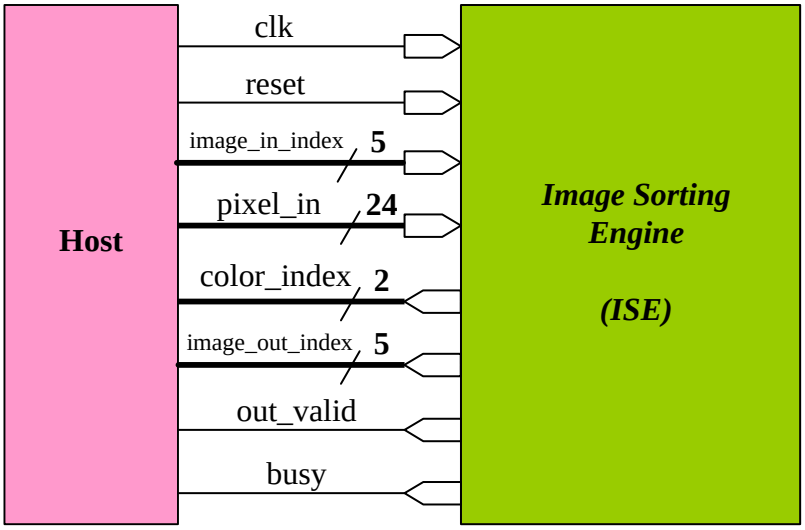
軟體環境及設計資料庫說明請參考附錄 F 與附錄 G。



圖一、Image Sorting Engine 之方塊圖

2.設計規格

2.1 系統方塊圖



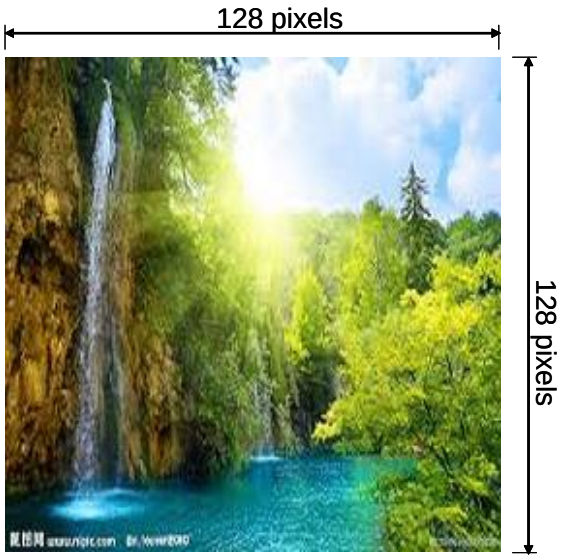
圖二、系統方塊圖

2.2 輸入/輸出介面

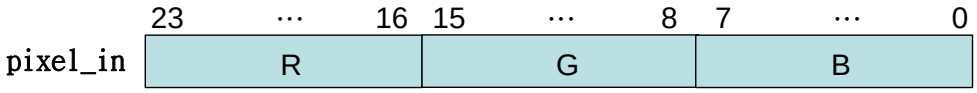
表 1 -輸入/輸出訊號

Signal Name	I/O	Width	Simple Description
clk	I	1	本系統為同步於時脈正緣之同步設計。 (註: Host 端採 clk ”正”緣時送資料。)
reset	I	1	高位準”非”同步(active high asynchronous)之系統重置信號。
image_in_index	I	5	ISE 影像之 index 值輸入匯流排。當 Host 端輸入 Pixel 影像訊號時，會透過此匯流排來指示，目前輸入的 Pixel 訊號是隸屬於哪張影像之 index 值。
pixel_in	I	24	ISE 影像 Pixel 訊號之匯流排。Host 端會透過此匯流排將影像的所有 Pixel 訊號進行輸入。每一個週期僅能輸入一組 Pixel 值，且輸入過的 Pixel 值無法再重複輸入。註：輸入訊號之 Pixel 結構與其順序，請參照圖四、圖五所示。
busy	O	1	ISE 忙碌之控制訊號。當為 High 時，表示系統正處於忙碌階段，告知 Host 端，暫時停止 pixel_in 資料的輸入；反之，當為 Low 時，表示告知 Host 端可繼續由 pixel_in 輸入資料。
out_valid	O	1	指示輸出資料為有效(Valid)之控制訊號。當為 High 時，表示目前輸出的資料為有效的；反之，當為 Low 時，表示目前輸出資料為無效的，即不被採用。

color_index	O	2	ISE 影像顏色類別之輸出匯流排。指示 Host 端，目前輸出的影像 index 值是屬於紅色類別則以 2'b00 輸出、綠色類別則以 2'b01 輸出、藍色類別則以 2'b10 輸出。 注意：每一個週期僅能輸出一筆資料。
image_out_index	O	5	ISE 影像所屬 index 值之輸出匯流排。當影像色彩分類與排序完成後，可透過此匯流排將各影像所屬之 index 值依序輸出。 注意：每一個週期僅能輸出一筆資料。



圖三、彩色影像資料



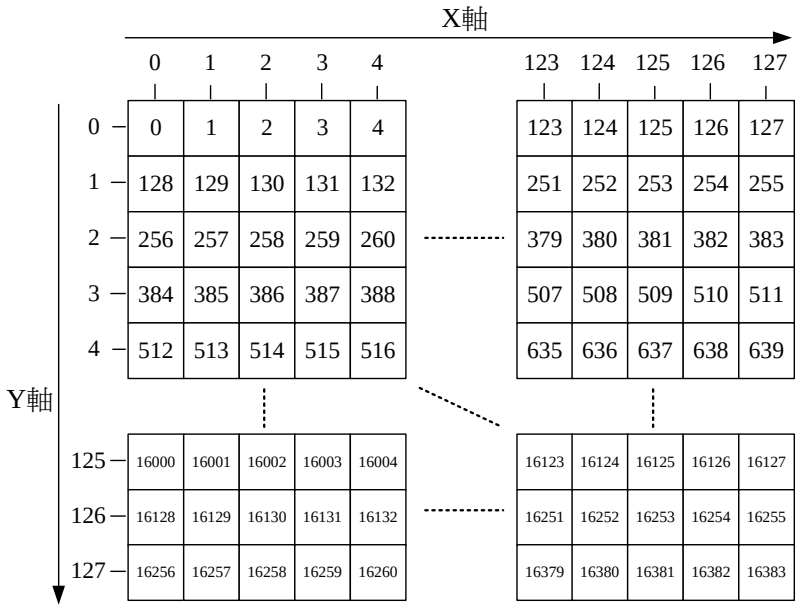
（本題pixel_in輸入訊號之結構 ）

圖四、彩色影像訊號之單一 Pixel 組合方式

2.3 系統描述

2.3.1 輸入端彩色影像訊號

圖三為一張彩色影像，影像中的每一點訊號稱為 Pixel，每個 Pixel 是由色彩三元素 R(Red)、G(Green)、B(Blue) 三基色分量的強弱組合來決定一個 Pixel 的顏色，例如：RGB 三基色分量(R，G，B) => (0，0，0)（即都最弱）時，該 Pixel 會呈現黑色，當 RGB 三基色分量(R，G，B) => (255，255，255)（即都最強）時，該 Pixel 會呈現白色，因此調整 RGB 三基色分量的值，可以調出各式各樣的顏色，本題 R、G、B 皆以 8 位元(即訊號強度為 0 ~ 255)表示，故彩色影像訊號的每個 Pixel 共計 24 位元，如圖四所示。本題影像尺寸為固定 128x128 個 Pixels，其輸入順序係由左到右，由上到下的順序(即如圖五所示，影像訊號輸入順序按照方格中之編號 0, 1, 2, 3, 4, ……, 16382, 16383)，循序輸入至參賽者的 ISE 電路中。



圖五、影像座標與影像輸入掃描順序圖

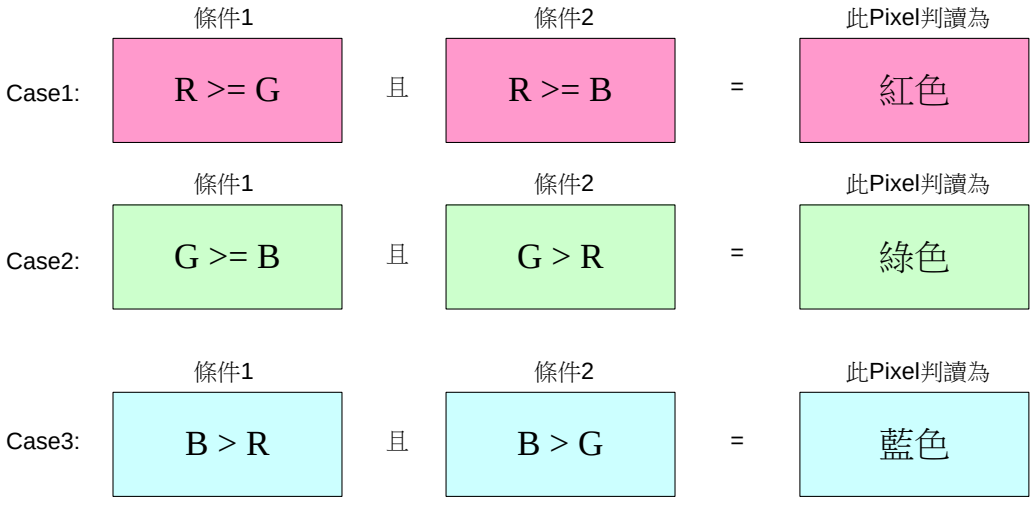
注意：

1. 任何一個 Pixel 之影像訊號只能讀取一次，並無反覆讀取之功能
2. 讀取過程中，參賽者若想暫停影像訊號輸入，可以透過 busy 控制訊號設為 High 即可，待處理適宜後，再將 busy 控制訊號設為 Low 即可繼續進行影像資料之讀取。

本題 Host 端會固定輸入 32 張彩色影像訊號，輸入的第一張影像 ISE 電路之 image_in_index 會輸入 00，第二張會輸入 01，….，最後一張影像會輸入 31。

2.3.2 ISE 電路運算方法

ISE 電路運算方法，首先要先判斷各影像 Pixel 是屬於紅色、綠色、藍色，可依據圖六法則來判斷。



圖六、單一 Pixel 顏色判讀方法

如前文所述，一張影像有 128x128 個 Pixels，當每個 Pixel 顏色皆判讀完後，需依照其數量之最多者，歸納該影像是屬於紅色類別或綠色類別或藍色類別。

分類法則：

a. 當紅色Pixel的數量在同一張影像中佔最多，該影像即分類為紅色類別。
b. 當綠色Pixel的數量在同一張影像中佔最多，該影像即分類為綠色類別。
c. 當藍色Pixel的數量在同一張影像中佔最多，該影像即分類為藍色類別。

例如：
一張影像128x128 pixels, 紅色佔10000個, 綠色佔3500個, 藍色佔2884個

=> 此影像 R_ratio = 10000 / 16384 = 0.6103
此影像 G_ratio = 3500 / 16384 = 0.2136
此影像 B_ratio = 2884 / 16384 = 0.1760

本範例R_ratio為最大, 因此本張影像歸納為紅色類別!

分類後之影像，需將其影像按照暗紅色...排列到亮紅色，暗綠色...排列到亮綠色，暗藍色...排列到亮藍色。其排列方法，本題規定如下：

暗/亮色判斷與排列法則：

a. 當一張影像紅色Pixel的比例佔最重，此張影像即歸納為紅色類別，此時將該張影像歸類為紅色pixels之強度加總，求其該張影像之紅色平均強度，其餘同樣歸類為紅色的影像，也依照此作法，求其平均強度，最後便可將同屬於紅色類別之所有影像，依照平均強度數值，由低至高排列，即可排列不同張紅色類別的影像，從暗紅...排列至亮紅。
b. 綠色也依上述作法，從暗綠...排列至亮綠。
c. 藍色也依上述作法，從暗藍...排列至亮藍。

2.3.3 ISE 電路輸出順序

ISE 電路輸出順序為：

a. 紅色類別影像 index 值，按照暗紅色...排列到亮紅色，依序輸出。
b. 綠色類別影像 index 值，按照暗綠色...排列到亮綠色，依序輸出。
c. 藍色類別影像 index 值，按照暗藍色...排列到亮藍色，依序輸出。

每一筆影像 index 值輸出時，需同時透過 ISE 電路之 color_index 腳位告知 Host 端，目前輸出的影像 index 是屬於紅色類別的給定 2'b00，屬於綠色類別的給定 2'b01，屬於藍色類別的給定 2'b10。

注意：

1. 每一個週期只能輸出一組影像 index 值(image_out_index)與色彩類別 index 值(color_index)。
2. 當輸出的 image_out_index 及 color_index 為有效的，需將 out_valid 訊號設為 1，反之則設為 0。
3. 本題之紅、綠、藍三大類別影像，其數量未必會一樣多，也可能某一類別從缺。

例如:

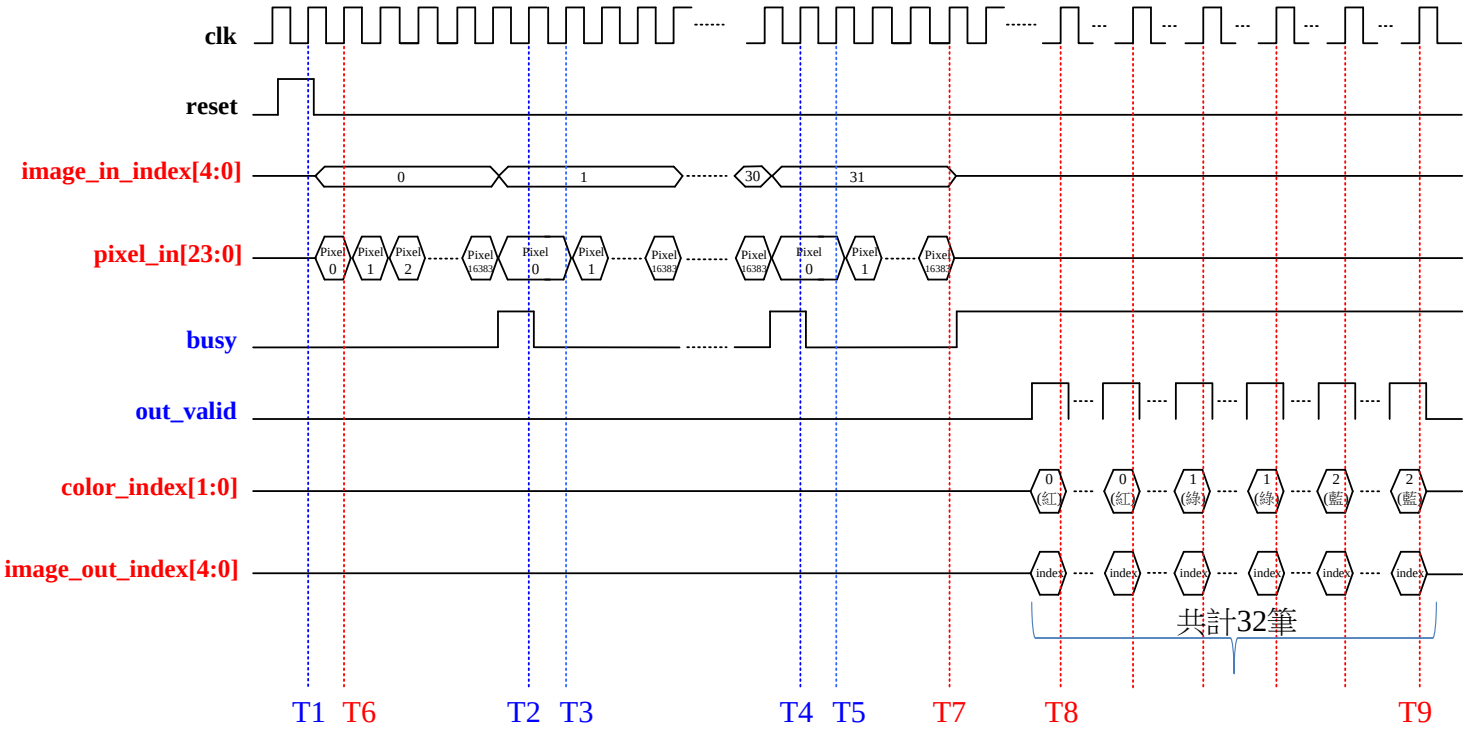
假設有六張 128x128pixels之影像，經統計後如下:

```
image 0 : (紅佔10000 pixels 綠佔 3500 pixels 藍佔 2884 pixels 紅色總訊號強度 1805420)
image 1 : (紅佔 1000 pixels 綠佔10000 pixels 藍佔 5384 pixels 綠色總訊號強度 2000000)
image 2 : (紅佔 2000 pixels 綠佔 4000 pixels 藍佔10384 pixels 藍色總訊號強度 2300000)
image 3 : (紅佔 8000 pixels 綠佔 6000 pixels 藍佔 2384 pixels 紅色總訊號強度 1444450)
image 4 : (紅佔 2000 pixels 綠佔 2000 pixels 藍佔12384 pixels 藍色總訊號強度 2450000)
image 5 : (紅佔 3000 pixels 綠佔 8000 pixels 藍佔 5384 pixels 綠色總訊號強度  680000)

=========================================================================

因此  image 0 => 紅色類別  (因 R_ratio=0.61 G_ratio=0.21 B_ratio=0.18 )
      image 1 => 綠色類別  (因 R_ratio=0.06 G_ratio=0.61 B_ratio=0.33 )
      image 2 => 藍色類別  (因 R_ratio=0.12 G_ratio=0.24 B_ratio=0.64 )
      image 3 => 紅色類別  (因 R_ratio=0.49 G_ratio=0.37 B_ratio=0.14 )
      image 4 => 藍色類別  (因 R_ratio=0.12 G_ratio=0.12 B_ratio=0.76 )
      image 5 => 綠色類別  (因 R_ratio=0.18 G_ratio=0.49 B_ratio=0.33 )

各影像之平均訊號強度:
      image 0 => R_avg = 1805420/10000 = 180.5420
      image 1 => G_avg = 2000000/10000 = 200.0000
      image 2 => B_avg = 2300000/10384 = 221.4946
      image 3 => R_avg = 1444450/ 8000 = 180.5563
      image 4 => B_avg = 2450000/12384 = 197.8359
      image 5 => G_avg =  680000/ 8000 =  85.0000

因此ISE電路輸出順序(由左到右)為:
         color_index:  2'b00      2'b00      2'b01      2'b01      2'b10      2'b10
     image_out_index:  5'b00000   5'b00011   5'b00101   5'b00001   5'b00100   5'b00010
```

注意：

1. 假若兩數間之平均訊號強度非常接近時，需考量比較至小數位數的部分。
2. 小數位數的精確度，需足以區別出兩數之大小為止。

2.4 ISE 電路時序規格



圖七、ISE 電路輸入之時序圖

1. ISE 電路初始化，Reset 一個 Cycle 的時間。
2. T1 時間點，reset 由於是非同步的因此 Host 端可以在 T1 時間點判斷 busy 訊號為 Low，隨即便送出第一筆 Pixel 訊號及該 Pixel 所屬之影像 index 值。
3. T2 時間點，假設參賽者需要暫停影像訊號輸入，可將 busy 拉為 High，Host 於 T2 時間點判斷 busy 為 High，pixel_in 的訊號便維持不變。T3 時間點，假設參賽者需要繼續獲得下一筆 Pixel，便將 busy 訊號設為 Low 即可！(T4、T5 時間點，與 T2、T3 的動作一樣，也只是假設需要暫停輸入的範例，參賽者未必需要跟著作暫停的動作)
4. T6 ~ T7 時間，Host 端總共輸入完 32 張影像(每張 16384 個 Pixels)後，pixel_in 與 image_in_index 便呈現高阻抗(High-impendence)狀態，此時 busy 訊號要為 Low 或 High 皆可，因 Host 端已完成所有 Pixel 輸入，無法再重新輸入已輸入過的 Pixel 值。
5. 當參賽者完成 ISE 電路運算後，欲將資料輸出，可以像 T8~T9 時間，將 out_valid 設為 High，告知 Host 端有資料輸出，隨即便將紅色類別的影像 index 值先做輸出、綠色類別隨後輸出，最後是藍色類別的影像 index 值。本題並未規定需要連續輸出，參賽者可自行控制 out_valid 控制訊號，決定何時要輸出就設為 High，不需輸出時設為 Low 即可。當 32 筆資料輸出後，系統便會自動結束模擬，若未達到 32 筆的資料輸出，系統將永無終止地模擬下去，請注意。

註：

1. 輸出順序雖然是紅色類別→綠色類別→藍色類別，但本題輸入的 32 張影像，”**未必**”三大類別的影像一定存在，請注意!**沒有的項目，請參賽者自行跳過。**

3.評分標準

評分方式會依設計完成程度，分成 A、B、C、D 四種等級，排名順序為 A>B>C>D，評分項目有兩個，分別為模擬時間、面積，主辦單位會依此兩項目做為同等級之評分。另外，請參賽者提供一組正確的週期時間(CYCLE TIME)給評分人員驗證本電路之正確性。

✧ **評分項目一：依"模擬時間"(Time)長短評分**

各參賽隊伍將 APR 完成後，執行 Gate-level Post-layout Simulation 模擬完後，會出現模擬時間，評分人員會以此模擬時間如下面範例，紀錄成 Time =4594041.84 ns 做評分。

```
Loading snapshot worklib.test:v ..................... Done
*Verdi3* Loading libsscore_ius131.so
*Verdi3* : Enable Parallel Dumping.
ncsim> source /usr/cad/cadence/INCISIV/cur/tools/inca/files/ncsimrc
ncsim> run
-----------------------------------------------------

Start to Send Pixel...
...
...
...
Send Pixel Over!
-----------------------------------------------------


Start to Run Output Compare....
-----------------------------------------------------

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

Congratulations! All data have been generated successfully!

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

Simulation complete via $finish(1) at time 4594041840 PS + 0
./testfixture3.v:156       #(`CYCLE/2); $finish;
ncsim> exit
```

註：Simulation Time 若不同則以最大值為準。

✧ **評分項目二：依"面積"(Area)大小評分**

各參賽隊伍將 APR 完成後，面積分析方法如下範例，請任選其一 APR 軟體做分析。

1. IC Compiler Report Area 範例:

icc_shell> get_attribute [get_die_area] bbox

{0.000 0.000} {581.15 574.90}

=> Area = 581.15 x 574.90 = 334103.13 um^2

2. Encounter Report Area 範例:

encounter > analyzeFloorplan

```
************************* Analyze Floorplan *************************
    Die Area(um^2)              : 334103.13
    Core Area(um^2)             : 322420.28
    Chip Density (Counting Std Cells and MACROs and IOs): 96.464%
    Core Density (Counting Std Cells and MACROs): 99.960%
    Average utilization         : 100.000%
    Number of instance(s)       : 28606
    Number of Macro(s)          : 0
    Number of IO Pin(s)         : 40
    Number of Power Domain(s) : 0
************************* Estimation Results *************************
```

=> Area = 334103.13 um^2

註： 指令 analyzeFloorplan 會破壞已完成 routing 的結果，執行該指令後絕對不可再存檔。

設計完成程度四種等級，如下：

✧ **等級 A： 達成"完成設計"之三項要求**

a、 功能正確，RTL 模擬與正確解答比對完全正確。

b、 完成 Synthesis，且 Gate-Level Pre-layout Simulation 結果正確。

c、 完成 APR，並達成 APR 必要項目，Gate-Level Post-layout Simulation 結果正確。

註：完成 APR 之必要項目

i. 只需做 Marco layout (即不用包含 IO Pad、Bonding Pad)。
ii. VDD 與 VSS Power Ring 寬度請各設定為 **2um**，只須做一組。
iii. 不需加 Dummy Metal。
iv. Power Stripe 務必至少加一組，其 VDD、VSS 寬度各設定為 **2um**。
(Power Stripe 垂直方向至少一組，水平方向可不加)
v. 務必要加 Power Rail (follow pin)。
vi. Core Filler 務必要加。
vii. APR 後之 GDSII 檔案務必產生。
viii. 完成 APR，DRC/LVS 完全無誤(見附錄 C 說明)。

註： 本題之 image_in_index 之腳位未必會用到，若因而引發 DRC/LVS 錯誤，可以忽略之。

等級 A 之評分方法：

Score = Area x Time

例如：

本範例(Encounter)為，

Score = Area x Time = 334103.13 x 4594041.84 = 1534883758094

註: Score 越小者，同級名次越好!

✧ **等級 B：已做到 APR，但等級 A 之"APR 必要項目"有部分不符合，DRC/LVS 錯誤總數量容許 5 個(含)以下**

此等級之成績計算方式如下：

Score = Area x Time x (DRC+LVS 的錯誤總數量)

註: Score 越小者，同級名次越好!

✧ **等級 C：僅完成合成，或做到 APR，但 DRC/LVS 錯誤總數量超過 5 個以上**

此等級之成績計算方式如下：

Score = Area x Time

註:

1. Score 越小者，同級名次越好!
2. 等級 C，視 APR 為 Fail，Area 以 Design Compiler 所 Report 的 Cell Area 為主。
3. 等級 C，視 APR 為 Fail，Time 以 Gate-level Pre-layout Simulation 為主。

✧ **等級 D：未達成前三等級者，**成績計算方式為 All RTL Simulation，比對結果之 error 總數量越少者，分數越高。

Score = Total error of All RTL Simulations

註:

1. 等級 D，Score 評分方式為所有模擬的 error 總數作相加。
2. 等級 D，視合成及 APR 皆為 Fail，Area、Timing 將不予考慮之。
3. 等級 D，只以 RTL Simulation 正確率為主，Score 越小者(即 error 越少)，同級名次越好。

附錄

附錄 A 為主辦單位所提供各參賽者的設計檔案說明；附錄 B 為主辦單位提供的測試樣本說明；附錄 C 為設計驗證說明；附錄 D 為評分用檔案，亦即參賽者必須繳交的檔案資料；附錄 E 則為設計檔案壓縮整理步驟說明；附錄 F 中說明本次競賽之軟體環境；附錄 G 中說明本次競賽使用之設計資料庫。

附錄 A 設計檔(For Verilog)

1. 下表為主辦單位所提供各參賽者的設計檔

表 3、設計檔案說明

檔名	說明
ISE.v	參賽者所使用的設計檔，已包含系統輸/出入埠之宣告。
testfixture1.v testfixture2.v testfixture3.v	本題共計三個 TestBench，每組測試皆為 32 張 128x128 pixels 之影像。
in_pattern1.dat in_pattern2.dat in_pattern3.dat	ISE 電路之 Input Pattern 檔案。每組 Input Pattern 檔案，皆有提供影像 index 值、影像座標值及各 Pixel 之 RGB 訊號值(每組 Pixel 皆提供十六進制與十進制兩種表示法，以方便 Debug 用!)
out_golden1.dat out_golden2.dat out_golden3.dat	ISE 電路之 Output Golden Pattern 檔案。每組 Output Golden Pattern 檔案，皆有提供輸出影像之顏色分類 index 值及影像 index 值(每組輸出皆提供十六進制與十進制兩種表示法，以方便 Debug 用!)
.synopsys_dc.setup	使用 Design Compiler 作合成或 IC Compiler Layout 之初始化設定檔。參賽者請依 Library 實際擺放位置，自行修改 Search Path 的設定。**注意：無論合成或 APR，只需使用 worst case library。**
ISE_DC.sdc	Design Compiler 作合成之 Constraint 檔案。參賽者可依需求自行修改 cycle 的設定，**若需使用到多個 Clock Domain 的設計，可自行補上需要用到的 Constraints**，其餘環境參數請勿更改。
ISE_APR.sdc	Encounter 或 IC Compiler 作 Layout 之 Constraint 檔案。參賽者可依需求自行修改 cycle 的設定，**若需使用到多個 Clock Domain 的設計，可自行補上需要用到的 Constraints**，其餘環境參數請勿更改。

請使用 ***ISE.v***，進行 ISE 電路之設計。其模組名稱、輸出/入埠宣告如下所示：

```
`timescale 1ns/10ps

module   ISE( clk, reset, image_in_index, pixel_in, busy, out_valid, color_index, image_out_index);
input              clk;
input              reset;
input    [4:0]     image_in_index;
input    [23:0]    pixel_in;
output             busy;
output             out_valid;
output   [1:0]     color_index;
output   [4:0]     image_out_index;

endmodule
```

2. 本題有三組 Pattern 要測試：testfixture1.v, testfixture2.v, testfixture3.v，其會用到之相關檔案已設定完成，請參賽者注意檔案(in_pattern.dat、out_golden.dat)擺放之路徑即可！

3. 本題所提供的 Test Bench 檔案，有多增加幾行特別用途的敘述如下：

```
`define   End_CYCLE   100000000
`define   SDFFILE       "./ISE_syn.sdf"
`ifdef   SDF
    initial   $sdf_annotate(`SDFFILE, ISE);
`endif
```

註：

1. End_CYCLE 預設 100 百萬個 Cycles，其目的可以防止參賽者因電路有錯，模擬陷入無窮回圈卻白白浪費時間空等之問題，倘若參賽者確定模擬真的需要超過 100 百萬個 Cycles 以上，請自行加大此 Cycle 數。
2. SDF 檔案，請自行修改 SDF 實際檔名及路徑後再模擬。
3. 在 Test Bench 中，主辦單位提供`ifdef SDF 的描述，其目的是讓本 Test Bench 可以作為 RTL 模擬、合成後模擬與 Layout 後模擬皆可使用。注意：當參賽者在合成或 Layout 後模擬，請務必多加一個參數"+define+SDF"，方可順利模擬。

例如：當合成後，使用 NC-Verilog 模擬第一組樣本，在 UNIX 下執行下面指令

```
> ncverilog   +ncmaxdelays   testfixture1.v   ISE_syn.v   –v   tsmc13_neg.v   +define+SDF
                +access+rw
```

註：使用 Encounter 作 APR 之參賽者，APR 後的模擬務必加上+ncmaxdelays 參數，請注意！

附錄 B 測試樣本

本題所提供之三組測試樣本，皆為 32 張 128 x 128 Pixels 之影像，影像內容如下：

測試樣本 1: (各影像 Pixel 值與 index 值，已存放於 in_pattern1.dat 中)

上面所列之影像訊號，已轉存成 in_pattern1.dat，其內容如下：

```
00_D1_9F_72            //Image Index=00  y=000  x=000 => Pixel(R=209  G=159  B=114)
00_D5_A4_6E            //Image Index=00  y=000  x=001 => Pixel(R=213  G=164  B=110)
00_C4_91_56            //Image Index=00  y=000  x=002 => Pixel(R=196  G=145  B=086)
00_BC_88_55            //Image Index=00  y=000  x=003 => Pixel(R=188  G=136  B=085)
00_94_5B_3E            //Image Index=00  y=000  x=004 => Pixel(R=148  G=091  B=062)
00_C5_8B_7A            //Image Index=00  y=000  x=005 => Pixel(R=197  G=139  B=122)
00_D6_97_88            //Image Index=00  y=000  x=006 => Pixel(R=214  G=151  B=136)
00_C0_83_78            //Image Index=00  y=000  x=007 => Pixel(R=192  G=131  B=120)
00_A0_65_6F            //Image Index=00  y=000  x=008 => Pixel(R=160  G=101  B=111)
00_C0_83_5C            //Image Index=00  y=000  x=009 => Pixel(R=192  G=131  B=092)
00_D0_92_52            //Image Index=00  y=000  x=010 => Pixel(R=208  G=146  B=082)
00_C4_88_60            //Image Index=00  y=000  x=011 => Pixel(R=196  G=136  B=096)
00_6E_3E_28            //Image Index=00  y=000  x=012 => Pixel(R=110  G=062  B=040)
00_F5_CB_AE            //Image Index=00  y=000  x=013 => Pixel(R=245  G=203  B=174)
00_A7_75_54            //Image Index=00  y=000  x=014 => Pixel(R=167  G=117  B=084)
00_AF_77_5D            //Image Index=00  y=000  x=015 => Pixel(R=175  G=119  B=093)
   :
```

註：資料左側為十六進制，每行開頭第一個"_" 符號，其左側為影像 index 值，右側為 RGB 訊號。

註：資料右側為左側數值之十進位數值，為方便參賽者 debug，每個 Pixel 都有加上座標軸。

測試樣本 1 之輸出結果：**(有關 ISE 細項運算，可以參考 2.3.3 之範例)**

```
0_1A            //Color Category(Red=0, Green=1, Blue=2):0  Image Index:26
0_1F            //Color Category(Red=0, Green=1, Blue=2):0  Image Index:31
0_02            //Color Category(Red=0, Green=1, Blue=2):0  Image Index:02
0_00            //Color Category(Red=0, Green=1, Blue=2):0  Image Index:00
0_1E            //Color Category(Red=0, Green=1, Blue=2):0  Image Index:30
0_0A            //Color Category(Red=0, Green=1, Blue=2):0  Image Index:10
0_05            //Color Category(Red=0, Green=1, Blue=2):0  Image Index:05
0_0F            //Color Category(Red=0, Green=1, Blue=2):0  Image Index:15
1_06            //Color Category(Red=0, Green=1, Blue=2):1  Image Index:06
1_07            //Color Category(Red=0, Green=1, Blue=2):1  Image Index:07
1_16            //Color Category(Red=0, Green=1, Blue=2):1  Image Index:22
1_01            //Color Category(Red=0, Green=1, Blue=2):1  Image Index:01
1_12            //Color Category(Red=0, Green=1, Blue=2):1  Image Index:18
1_03            //Color Category(Red=0, Green=1, Blue=2):1  Image Index:03
1_14            //Color Category(Red=0, Green=1, Blue=2):1  Image Index:20
 :
```

註：資料左側為十六進制，每行"_" 符號，其左側為顏色類別之 index 值，右側為影像 index 值號。
註：資料右側為左側數值之十進位數值。

測試樣本 2: (各影像 Pixel 值與 index 值，已存放於 in_pattern2.dat 中)

測試樣本 3: (各影像 Pixel 值與 index 值，已存放於 in_pattern3.dat 中)

附錄 C 設計驗證說明

參賽者繳交資料前應完成 RTL，Gate-Level 與 Physical 三種階段驗證，以確保設計正確性。注意：每組限定只能使用 1 license, 勿使用 Multi-CPU。

- RTL 與 Gate-Level 階段：參賽者必須進行 RTL simulation 及 Gate-Level simulation，模擬結果必須於參賽者提供之 CYCLE 數值下，功能完全正確。
- Physical 階段，包含三項驗證重點：

1. 依主辦單位各項要求，實現完整且正確的 layout (詳細之各項要求，請見評分標準)。

2. 完成 post-layout simulation：參賽者必須使用 P&R 軟體**寫出之 netlist 檔與 sdf 檔完成 post-layout gate-level simulation**，以下分為 IC Compiler、Encounter 兩種軟體說明 netlist 與 sdf 寫出步驟。

   i. 使用 Synopsys IC Compiler 者，執行步驟如下：

   在 IC Compiler 主視窗底下點選

   “ File > Export > Write SDF… ”

Specify Version	Version 2.1
Instance	空白即可
File name	ISE_pr.sdf
Significant digits	2

   按 OK。

   對應指令： write_sdf -version 2.1 ISE_pr.sdf

   “ File > Export > Write Verilog… ”

   先按 Default

Output verilog file name	ISE_pr.v
Output physical only cells	disable
Wire declaration	enable
Backslash before Hierarchy Separator	Enable
All other options	Default value

   按 OK。

   ii. 使用 Cadence Encounter 者，執行步驟如下：

   在 Encounter 視窗下點選：

   “ File → Save → Netlist… ”

Netlist File	ISE_pr.v
All other options	Default value

   按 OK。

   “ Timing → Extract RC… ” ，按 OK。

“ Timing → Write SDF… ”

Ideal Clock	Disable
SDF Output File:	ISE_pr.sdf

按 OK。

3. **完成 DRC 與 LVS 驗證：參賽者必須以其所使用之 P&R 軟體內含之 DRC 與 LVS 驗證功能完成 DRC 與 LVS 驗證，以下分為 IC Compiler、Encounter 兩種軟體說明執行步驟。**

i. 使用 Synopsys IC Compiler 者，驗證 DRC 與 LVS 步驟如下：

在 IC Compiler Layout 視窗底下點選

“ Route > Verification > DRC …”

Read child cell from	Cell view
All other options	Default value

按 OK。

將跳出 Error Browser 視窗，請參賽者自行查看是否有錯，若有請自行修改 Layout 到 0 個 Violation 為止。

“ Route > Verification > LVS …”

Pins not connected to a wire segment(Floating port)	disable
All other options	Default value

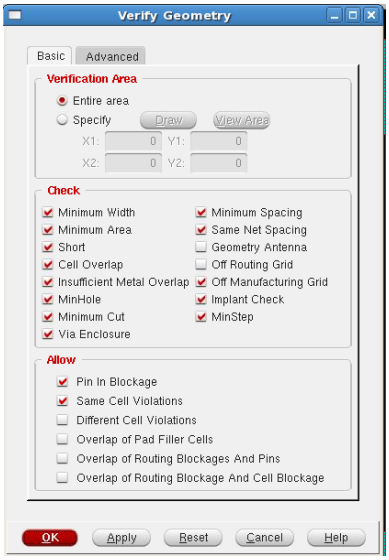
按 OK。

將跳出 Error Browser 視窗， 檢查看看是否有錯，若有請自行修正到 0 個 Violation 為止。

ii. 使用 Cadence Encounter 者，驗證 DRC 與 LVS 步驟如下：

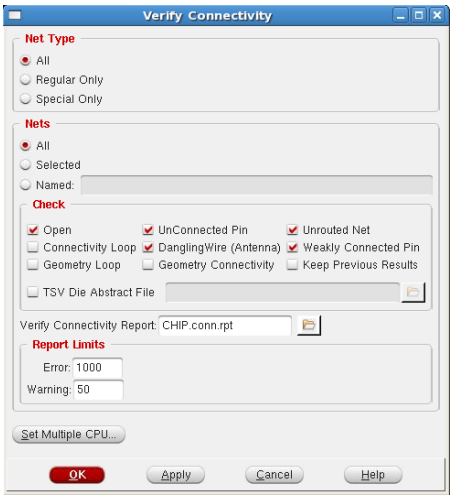
在 Encounter 視窗下點選

1. DRC 驗證： 請選**“Verify → Verify Geometry… ”** Default 值，按 OK。



註: 若 DRC 有發生錯誤，請選**“Tools → Violation Browser… ”**查明原因。

2. LVS 驗證： 請選**“Verify → Verify Connectivity… ”** Default 值，按 OK。



註: 若 LVS 有發生錯誤，請選**“Tools → Violation Browser… ”**查明原因。

附錄 D 評分用檔案

評分所須檔案可以下幾個部份：(1)RTL design，即各參賽隊伍對該次競賽設計的 RTL code，若設計採模組化而有多個設計檔，請務必將合成所要用到的各 module 檔放進來，以免評審進行評分時，無法進行模擬；(2)Gate-Level design，即由合成軟體所產生的 gate-level netlist，以及對應的 SDF 檔；(3)Physical design，使用 Synopsys IC Compiler 者，請記得將整個 Milkyway Library 等相關的 design database，壓縮成一個檔案。使用 Cadence Encounter 者，請將 Encounter 相關的 design database（包含.enc 檔案與 and .enc.dat 目錄），壓縮成一個檔案。壓縮的檔案格式如下：假設參賽者的 design database 目錄名稱為”your_lib”，請執行底下的 UNIX 指令，最後可以得到”your_name.tar”的檔案。

> ***tar cvf your_name.tar your_lib***

在執行以上的指令之前，請確定將你使用的 P&R Tool 儲存後關閉，再執行上述的指令，否則在壓縮的過程會出現錯誤。

表 4

RTL category		
Design Stage	*File*	*Description*
N/A	`N/A`	Design Report Form
RTL Simulation	`*.v or *.vhd`	Verilog (or VHDL) synthesizable RTL code
Gate-Level category		
Design Stage	*File*	*Description*
Pre-layout Gate-level Simulation	`*_syn.v`	Verilog gate-level netlist generated by Synopsys Design Compiler
	`*_syn.sdf`	Pre-layout gate-level sdf
Physical category		
Design Stage	*File*	*Description*
P&R	`*.tar`	**archive of the design database directory**
	`*.gds`	GDSII layout
	`DRC/LVS report`	不用儲存 DRC/LVS Report 檔案!只需在 Design Report Form 上填寫 DRC/LVS 錯誤總數量即可。(目標要做到 0 個錯誤!)
Post-layout Gate-level Simulation	`*_pr.v`	Verilog gate-level netlist generated by Cadence Encounter or Synopsys IC Compiler
	`*_pr.sdf`	Post-layout gate-level sdf

附錄 E 檔案整理步驟

當所有的文件準備齊全如表 4 所列，請按照以下的步驟指令，提交相關設計檔案，將所有檔案複製至同一個資料夾下，步驟如下:

1. 在自己的 home directory 建立一個新目錄，名稱叫做"**result**" 例如：
   > ***mkdir ~/result***
2. 將附錄 D 要求的檔案複製到 result 這個目錄。例如:
   > ***cp ISE.v ~/result/***
   > ***cp ISE_syn.v ~/result/***
   . . . . .
3. 在 Design Report Form 中，填入所需的相關資訊。

附錄 F 軟體環境

1. 使用者登入後自動會設定好以下軟體環境：

Vendor	Tool	Executable
Cadence	Virtuoso *1	icfb
	Composer	icfb
	NC-Verilog	ncverilog
	SOC Encounter	encounter
Synopsys	Design Compiler	dv, dc_shell
	VCS-MX	vcs
	IC Compiler	icc_shell -gui
	Hspice	hspice
	Cosmos Scope *1	scope
	Custom Explorer *1	wv
	Laker *1	laker
	Laker ADP*1	adp
	Verdi *1	verdi, nWave, nLint
Mentor	Calibre *3	calibre
	ModelSim	vsim
Utility	vi	vi, vim
	gedit	gedit
	nedit	nedit
	pdf reader	acroread
	calculate	gnome-calculator, bc -l
	gcc	gcc

EDA 軟體所須使用的 license 皆已設定完成，不須額外設定

*1 該軟體限定使用 1 套 license

*3 該軟體限定使用 3 套 license

附錄 G 設計資料庫

設計資料庫位置： **/cad/icc2015/CBDK_IC_Contest_v2.1**

目錄架構

ICC/		
	tsmc13gfsg_fram/	ICC core library
	tsmc13_CIC.tf	ICC technology
	macro.map	layer mapping file
	tluplus/	
	t013s8mg_fsg_typical.tluplus	t13 tluplus file
	t013s8mg_fsg.map	t13 tluplus mapping file
SOCE/		
	lef/	
	tsmc13fsg_8lm_cic.lef	LEF for core cell
	antenna_8.lef	LEF for antenna
	lib/	
	slow.lib	worst case for core cell
	streamOut.map	Layout map for GDSII out
SynopsysDC/		
	db/	
	slow.db	Synthesis model (slow)
	lib/	
	slow.lib	timing and power model
Verilog/		
	tsmc13_neg.v	Verilog simulation model
Phantom/		
	tsmc13gfsg_fram.gds	Standard Cell GDSII file

Design Report Form

登入帳號(login-id)		
RTL category		
Design Stage	*Description*	*File Name*
RTL Simulation	使用之 HDL 名稱 (請填入 Verilog 或 VHDL)	
RTL Simulation	RTL 檔案名稱 (RTL Netlist file name)	
Gate-Level category		
Design Stage	*Description*	*File Name*
Pre-layout Gate-level Simulation	Gate-Level 檔案名稱 (Gate-Level Netlist file name)	
	Pre-layout sdf 檔案名稱	
	Gate-Level simulation, 所使用的 CYCLE Time (請確定模擬功能正確)	( ) ns
Physical category		
Design Stage	*Descritpion*	*File Name or Value*
P&R	使用之 P&R Tool (請填入 IC compiler 或 Encounter)	
	設計資料庫檔案名稱(Library name) (ICC: Milkyway Library Name, Encounter: xxx.enc.dat)	
	DRC 錯誤總數量 (ex: 0 個)	
	LVS 錯誤總數量 (ex: 0 個)	
	佈局檔檔案名稱(GDSII file name)	
	佈局面積(Layout **Area**)	( ) um^2
Post-layout Gate-level Simulation	Gate-Level 檔案名稱 (Gate-Level Netlist file name)	
	Post-layout sdf 檔案名稱	
	Post-layout Simulation 所使用的 CYCLE Time (請確定模擬功能正確) (請寫出三組模擬中的最大值)Ex: 2ns	
	Post-layout Simulation Time (Simulation Time,ex: 4594041 ns) **Time = ?**(請寫出三組模擬中的最大值)	
Over All	最後完成之等級?(ex: 等級 A)	
其他說明事項(Any other information you want to specify:(如設計特點 ...) 如寫不下可寫於背面		