

# 2010 University/College IC Design Contest

## Cell-Based IC Design Category for Undergraduate Student

### *Adaptive Threshold Engine*

#### 1.問題描述

請完成一 Adaptive Threshold Engine(後文以 **ATE** 表示)的電路設計。如圖一，本 ATE 電路的功能是將一灰階影像分離出前景影像，系統方塊圖如圖二，其詳細規格將描述於後。

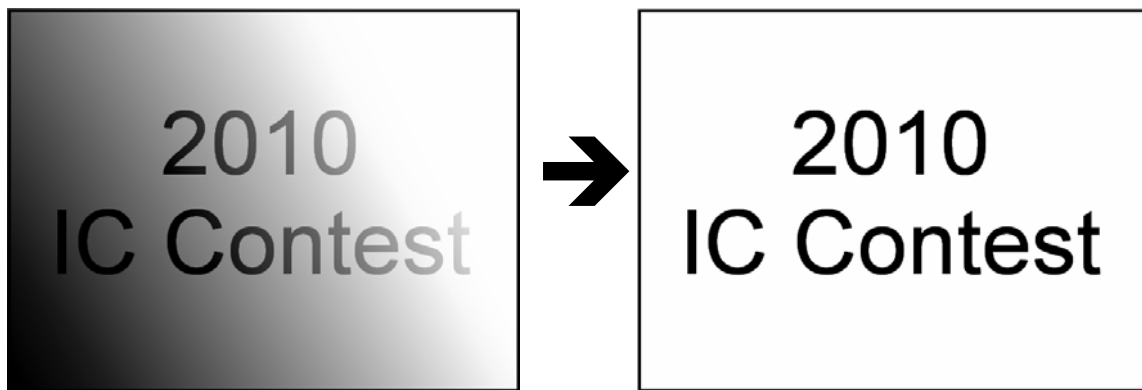
本電路各輸入輸出信號的功能說明，請參考表一。每個參賽隊伍必須根據下一節所給的設計規格及附錄 A 中的測試樣本完成設計驗證。

本次 IC 設計競賽比賽時間為上午 09:00 到下午 21:00。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/icc2010/ucb/icc2010ucb.tar**，請執行以下指令取得測試樣本：

```
tar xvf /usr/cad/icc2010/ucb/icc2010ucb.tar
```

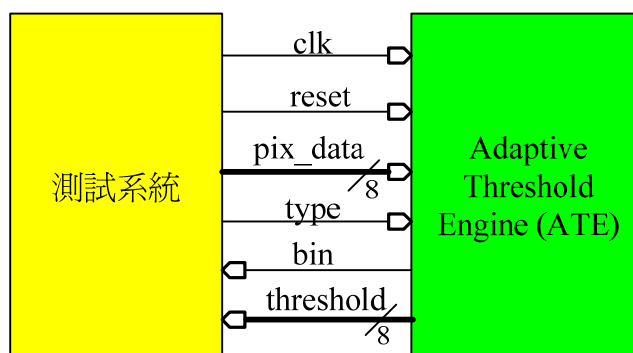
**軟體環境及設計資料庫說明請參考附錄 E 與附錄 F。**



圖一、Adaptive Threshold Engine 電路功能

## 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。
reset	I	1	高位準非同步(active high asynchronous)之系統重置信號。
pix_data	I	8	影像資料
type	I	1	輸入影像大小定義
bin	O	1	經 ATE 處理後輸出單點資料
threshold	O	8	單一區域之 threshold 值

### 2.3 系統描述

#### 2.3.1 影像輸入

本 ATE 所接受的輸入影像有兩種大小，以 type 設定，當 type=0 時，輸入影像大小為 48x32(6x4 區塊)，當 type=1 時，輸入影像大小為 528x512(66x64 區塊)；一個區塊固定為 8x8 個點，如圖三所示為 type=0 時的輸入影像，48x32 的影像共分為 6x4 個區塊，每個區塊是 8x8 個點。影像輸入順序是以區塊為單位依序輸入，如圖三之區塊編號順序。

執行 Threshold 處理時，是以一個區塊作為單位，而最左邊及最右邊兩行不需處理，一律輸出 0 (bin 及 threshold 皆是)，如圖三之 0, 6, 12, 18 區塊及 5, 11, 17, 23 區塊皆不需處理，其餘 16 個區塊必須個別計算出 threshold 數值，然後輸出 thresholding 之後的結果。

0	1	2	3	4	5
6	7	8	9	10	11
12	13	14	15	16	17
18	19	20	21	22	23

圖三、輸入影像區塊示意圖

### 2.3.2 Adaptive Threshold 做法

所謂 thresholding 目的是將一灰階影像的前景及背景區分出來，對於該影像區塊的每一個點，若大於或等於該區塊門檻值(threshold)則輸出1，反之則輸出0。而 Adaptive threshold 表示此 threshold 是經由計算所得。

本 ATE 電路的 threshold 計算方法採用單一 8x8 區塊中之最大數值及最小數值的平均值，即  $threshold = (Max + Min) / 2$ ，若 threshold 有小數，則採無條件進位。

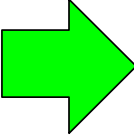
對於每一點輸出，其計算公式如下：

$bin=1$  if  $pix\_data \geq threshold$

$bin=0$  if  $pix\_data < threshold$

以圖四左圖為例，該區塊之最大值為 192，最小值為 48，則  $threshold = (192 + 48) / 2 = 120$ ，因此輸出為圖四之右圖。

48	48	150	138	150	150	150	150
128	51	51	128	138	138	150	150
138	51	51	48	48	128	150	150
138	138	128	48	48	131	138	150
150	150	138	128	48	69	69	69
150	150	150	138	128	69	69	69
192	150	150	150	138	69	69	120
192	192	192	150	150	120	120	120



0	0	1	1	1	1	1	1
1	0	0	1	1	1	1	1
1	0	0	0	0	1	1	1
1	1	1	0	0	1	1	1
1	1	1	1	0	0	0	0
1	1	1	1	1	0	0	0
1	1	1	1	1	0	0	1
1	1	1	1	1	1	1	1

圖四、單一區塊 thresholding 範例

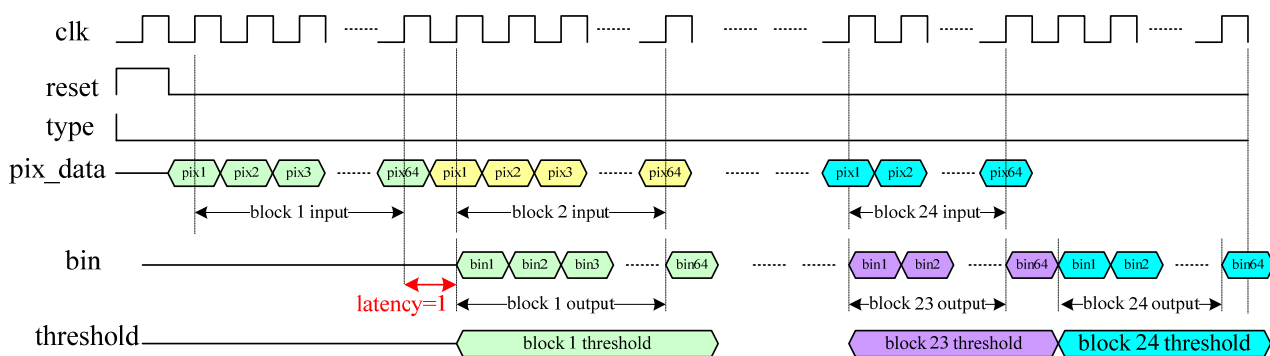
## 2.4 時序規格

### 2.4.1 輸出與輸出順序

測試電路一開始會進行 reset，reset 結束後，以連續不中斷的方式將整個影像內容輸入，輸入之區塊順序是由左至右，由上至下；在單一區塊中，亦是以由左而右，由上而下輸入每一個點的數值。輸出時之順序同輸入之順序，但輸出會比輸入晚一個區塊加上一輸出延遲時間(latency)，亦採不間斷連續輸出。

如圖五為當輸出延遲設定為 1 週期(latency=1)時的情形，當測試電路輸入完第一區塊內容時，ATE 在下一個 cycle 將第一區塊的計算結果送出。

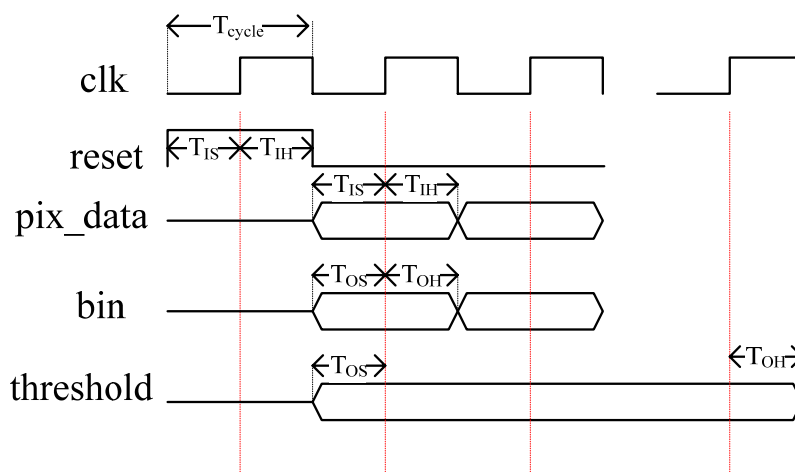
latency 以 cycle 為單位，參賽者可行定義自己設計電路的 latency，設定方式見附錄 A 第 2 項



圖五、輸入與輸出順序

### 2.4.2 輸出與輸出時序

本電路之輸入與輸出時序規格如下：



圖六、時序定義

表 2-時序定義說明

Symbol	Description	Value
T <sub>cycle</sub>	時脈週期	使用者定義
T <sub>IS</sub>	setup time (訊號送出到時脈正緣觸發的時間)	1/2 時脈週期
T <sub>IH</sub>	hold time(時脈正緣觸發到訊號移除時間)	1/2 時脈週期
T <sub>OS</sub>	setup time (從輸出有效資料到時脈正緣觸發的時間)	1/2 時脈週期
T <sub>OH</sub>	hold time (從時脈正緣觸發到輸出資料無效的間間)	1/2 時脈週期

### 3. 評分標準

評分人員將驗證參賽者繳交之設計，通過驗證者以”完成設計”所列之公式評分。但若通過完成設計的組數太少，CIC 亦將根據繳交資料依據 ”完成部分設計”所列分析各組之完成度：

#### 1. 完成設計

完成設計者意指：

- a、 RTL 通過所有測試樣本模擬
- b、 完成 Synthesis，且 gate-level netlist 通過所有測試樣本模擬

通過以上條件者以下列所列之公式評分。

$$\text{Cost} = \text{Area} * \text{Time} \quad (1)$$

Area：synthesis所report之面積大小(單位：um<sup>2</sup>)

Time：對測試樣本 2 執行 gate-level 模擬之實際模擬時間。

Cost 值越低，所得分數越高。

註：參賽者請自行找到最佳成績的 clock cycle 值，並填在 Design Report Form 裏；CIC 審核時，若發現 Design Report Form 未填寫 clock cycle，或是所填寫的值無法通過模擬，一律僅以 5ns、10ns、20ns 三種 clock cycle 進行測試與評分。

#### ✧ 做 APR 的隊伍可加分：

參賽隊伍若有達成”完成設計”之三項要求，可以考慮多作品片佈局(APR)，假如完成 APR(DRC、LVS 驗證無誤)，且 Gate-Level Post-layout Simulation 有達到完成設計之 a、b 項的要求，便可進行加分。加分方式如下。

$$\text{New\_cost} = \text{Area} * \text{Time} * 0.9 \quad (2)$$

Area：synthesis所report之面積大小(單位：um<sup>2</sup>)

Time：對測試樣本 2 執行 post-layout gate-level 模擬之實際模擬時間。

註：若 APR 後 Clock cycle 變長，則以比較(1)式及(2)式結果成績較佳者作為該組成績。

註：參賽者請自行找到最佳成績的 clock cycle 值，並填在 Design Report Form 裏；CIC 審核時，若發現 Design Report Form 未填寫 clock cycle，或是所填寫的值無法通過模擬，一律僅以 5ns、10ns、20ns 三種 clock cycle 進行測試與評分。

#### ✧ 實際模擬時間(Real Simulation Time)之補充：

範例: ATE Function1 的 Time 值

假設參賽者使用 ncverilog 之模擬結果如下：

Congratulations !  
Simulation Complete!!

Simulation complete via \$finish(1) at time 16025 NS + 0  
./testfixture2.v:172 \$finish;

在本例中，Time = 16025ns。

## 2. 完成部分設計

對於未完成 synthesis 設計者，共分為三個等級

I. synthesis 後之 gate-level 電路通過測試樣本 1 或測試樣本 2

II. RTL 電路通過所有測試樣本

III. RTL 電路通過測試樣本 1 或測試樣本 2

.....

## 附錄

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

### 附錄 A 設計檔(For Verilog or VHDL)

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
ate.v	參賽者所使用的設計檔，已包含系統輸/出入埠之宣告
testfixture.v	測試樣本檔。參賽者請自行調整本檔案定義之 CYCLE 及 LATENCY 值，確保在沒有 Setup/Hold Time 的問題下，花最少實際模擬時間，完成所有驗證測試。
tb1.map	測試樣本 1，輸入影像大小為 type1 大小
tb1.bin tb1.threshold	測試樣本 1 的 bin 及 threshold 結果
tb2.map	測試樣本 1，輸入影像大小為 type2 大小
tb1.bin tb1.threshold	測試樣本 2 的 bin 及 threshold 結果
.synopsys_dc.setup	使用 Design Compiler 作合成之初始化設定檔。參賽者請依 Library 實際擺放位置，自行填上 Search Path 的設定。
ate_syn.sdc	使用 Design Compiler 作合成之 sdc 檔。參賽者可自行調整最佳之 cycle 值。
ate_pr.sdc	APR 使用之 sdc 檔。參賽者可自行調整最佳之 cycle 值。



請使用 *ate.v*，進行 Adaptive Threshold Engine 之設計。其模組名稱、輸出/入埠宣告如下所示：

```
module ATE ( clk, reset, pix_data,type,bin,threshold);
    input    clk ;
    input    reset ;
    input    [7:0]  pix_data;
    input    type;
    output   bin;
    output   [7:0]  threshold;
endmodule
```

2. 更改電路 clock cycle 及 latency:

參賽者可自行更改 clock cycle 及 latency 設定使之滿合自己的設計，更改方式如下：

➤ 更改 *testfixture.v*

更改第 3 之 CYCLE 變數，及第 4 行之 LATENCY 變數，如下:

```
`define CYCLE 20 /*時脈週期更改為 20 ns */
`define LATENCY 1 /*輸出延遲為 1 cycle*/
```

➤ 更改 *ate\_syn.sdc* 及 *ate\_pr.sdc*

更改第 2 行 cycle 變數，如下

```
set cycle 20 #時脈週期更改為 20 ns
```

3. 比賽共提供 2 組測試樣本，參賽者可依下面範例來進行模擬:

➤ ncverilog 指令範例如下：

```
ncverilog testfixture.v ate.v +define+tb1
```

➤ 若使用 modelsim，則是在 compiler verilog 時，使用下面指令：

```
vlog testfixture.v +define+tb1
```

➤ 若使用 vcs，則是在 compiler verilog 時，使用下面指令：

```
vcs -R +v2k testfixture.v ate.v +define+tb1
```

➤ 上述指令中 *+define+tb1* 指的是使用第一組測試樣本模擬，若須使用其它測試樣本請自行修改此參數。以第二組測試樣本為例：*+define+tb2*。

4. dump 波形檔請參考下列指令：

➤ ncverilog 指令範例如下：

```
ncverilog testfixture.v ate.v +define+tb1+FSDB +access+r
```

➤ modelsim 使用者，請直接使用內建波形來進行除錯。

➤ vcs 指令範例如下:

```
vcs -R +v2k testfixture.v ate.v +define+tb1+FSDB -P
/usr/cad/spring_soft/verdi/cur/share/PLI/vcs_latest/LINUX/verdi.tab
/usr/cad/spring_soft/verdi/cur/share/PLI/vcs_latest/LINUX/pli.a
```

以上 3 行為同一行

5. gate-level 模擬請加 *+define+SDF*

## 附錄 B 設計驗證說明

參賽者繳交資料前應完成 RTL，Gate-Level 與 Physical (Optional，可以不作)三種階段驗證，以確保設計正確性。

- RTL 與 Gate-Level 階段：參賽者必須進行 RTL simulation 及 Gate-Level simulation，模擬結果必須於自行定義的系統時脈下，輸出結果正確且無 setup/hold time 的問題。
- Physical 階段，包含兩項驗證重點：
  1. 完成最後 layout，
    - i. Marco layout，不含 IO Pad。
    - ii. VDD 與 VSS power ring 寬度請各設定為 2um。

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、Astro、SOC Encounter 三種軟體說明 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	ate_pr.sdf
Significant digits	2

按 。

對應指令： write\_sdf -version 2.1 ate\_pr.sdf

“File > Export > Write Verilog...”

先按

Output verilog file name	ate_pr.v
Output physical only cells	disable
Wire declaration	enable
Backslash before Hierarchy Separator	enable

按 。

- ii. 使用 Synopsys Astro 者，執行步驟如下：

在 Astro 視窗底下點選

**“Timing > SDF Out”**

Specify Version	Version 2.1
Operation Mode	Normal SDF
File Name	ate_pr.sdf

按 。

**“Cell > Hierarchical Verilog Out”**

Flattened Cell Name (.EXP .CEL)	ate.CEL
Enter File Name	ate_pr.v
No power/ground ports	Enable
No power/ground nets	Disable
Output bus as individual bits	Disable
No empty Cell Module Definitions	Enable
No Corner Pad Instances	Enable
No Pad Filler Cell Instances	Enable
No Core Filler Cell Instances	Enable
No Unconnected Cell Instances	Enable
No Unconnected Ports	Enable
Strip BackSlash Before Hierarchy Separator	Enable
No Diode Ports	Enable
Output Wire Declaration	Enable
Output 1'b1 for Power(VDD, vdd, ...) and 1'b0 for Ground(VSS, gnd, ...)	Enable
Generate macro definitions	Disable

按 。

- iii. 使用 Cadence SOC Encounter 者，執行步驟如下：

在 SOC Encounter 視窗下點選：

**“Design → Save → Netlist...”**

Netlist File	ate_pr.v
All other options	Default value

按 。

**“Timing → Calculate Delay...”**

存成 ate\_pr.sdf，按 。

註：如果發現 **Calculate Delay** 功能是灰色的(無法點選)，請先將目前結果存檔後離開 Encounter，再重新進入 Encounter 並 Restore 回原本 Design 即可。

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、Astro、SOC Encounter 三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 。

將跳出 Error Browser 視窗，請參賽者自行查看是否有錯，若有則自行修改 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 。

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 Synopsys Astro 者，驗證 DRC 與 LVS 步驟如下：

在 Astro 視窗底下點選

**“Verify > DRC ”**

List Error Summary Immediately	Enable
All other options	Default value

按 。

將跳出來的 DRC report 存成 DRC.report 檔。

**“Verify > LVS ”** Default 值，按 。

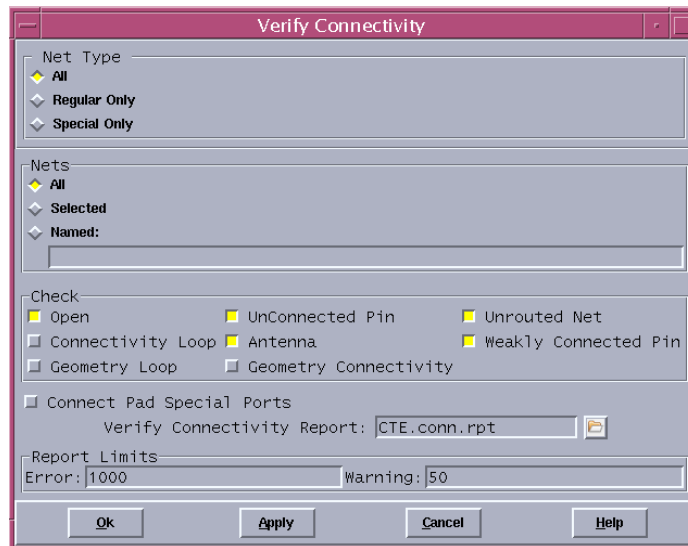
List Error Summary Immediately	Enable
All other options	Default value

將跳出來的 LVS report 存成 LVS.report 檔。

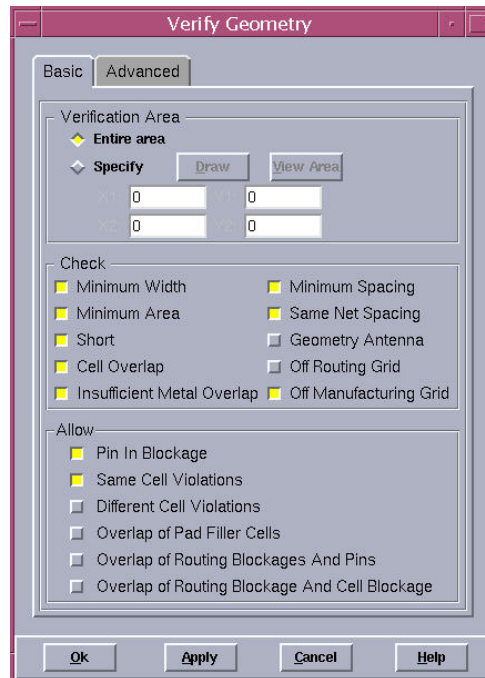
- iii. 使用 Cadence SOC Encounter 者，驗證 DRC 與 LVS 步驟如下：

在 SOC Encounter 視窗下點選

**“Verify → Verify Connectivity...”** Default 值，按 。



“Verify → Verify Geometry...” Default 值，按 **OK**。



“Verify → Violation Browser...”  
將 Verify 的結果存成 ate.viols.rpt

## 附錄 C 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/Astro者，請記得將整個相關的design library，壓縮成一個檔案。使用Cadence SOC Encounter者，請將SOC Encounter相關的design library（包含一個.enc檔及一個.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 library目錄名稱爲”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，再執行以上的指令，否則在壓縮的過程會出現錯誤。

表 3

<i>RTL category</i>		
<i>Design Stage</i>	<i>File</i>	<i>Description</i>
N/A	N/A	Design Report Form
RTL Simulation	*.v or *.vhd	Verilog (or VHDL) synthesizable RTL code
<i>Gate-Level category</i>		
<i>Design Stage</i>	<i>File</i>	<i>Description</i>
Pre-layout Gate-level Simulation	*_syn.v	Verilog gate-level netlist generated by Synopsys Design Compiler
	*_syn.sdf	Pre-layout gate-level sdf
<i>Physical category</i>		
<i>Design Stage</i>	<i>File</i>	<i>Description</i>
P&R	*.tar	<b>archive of the design library directory</b>
	*.gds	GDSII layout
	DRC/LVS report	For Astro : DRC.report ; LVS.report For SOC Encounter : CTE.viols.rpt
Post-layout Gate-level Simulation	*_pr.v	Verilog gate-level netlist generated by PR tool
	*_pr.sdf	Post-layout gate-level sdf

## 附錄 D 檔案整理步驟

當所有的文件準備齊全如表 3 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 C 要求的檔案複製到 result 這個目錄。例如：

```
> cp ate.v ~/result/
```

```
> cp ate_syn.v ~/result/
```

.....

3. 在 Design Report Form 中，填入所需的相關資訊。

## 附錄 E 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	icfb
	Composer	icfb
	Spectre	spectre
	NC-Verilog	ncverilog
	SOC Encounter	encounter
Synopsys	design vision	dv, dc_shell
	VCS	vcs
	Astro	Astro
	IC compiler	icc_shell -gui
	Hspice	hspice
	Cosmos Scope	scope
	Spice explorer	sx -w
Mentor	Calibre	calibre
	ModelSim	vsim
Spring Soft	Laker	laker
	Verdi	Verdi, nWave
Utility	vi	vi, vim, gvim
	joe	joe
	textedit	textedit
	nedit	nedit
	acroread	xpdf
	calculate	gnome-calculator, bc

EDA 軟體所須使用的 license 皆已設定完成，不須額外設定，且每組限定**每個軟體只能使**

用一套 license 。

## 附錄 F 設計資料庫

設計資料庫位置： /usr/cad/icc2010/CBDK\_IC\_Contest\_v2.0

### 目錄架構

Astro/	tsmc13gfsg_fram/ tsmc13_CIC.tf macro.map tluplus/ t013s8mg_fsg_typical.tluplus t013s8mg_fsg.map	Astro/ICC core library Astro/ICC technology layer mapping file  t13 tluplus file t13 tluplus mapping file
SOCE/	lef/ tsmc13fsg_8lm_cic.lef antenna_8.lef  lib/ fast.lib slow.lib typical.lib streamOut.map	LEF for core cell LEF for antenna  best case for core cell worst case for core cell typical case for core cell Layout map for GDSII out
SynopsysDC/	db/ fast.db slow.db typical.db  lib/ fast.lib slow.lib typical.lib	Synthesis model (fast) Synthesis model (slow) Synthesis model (typical)  timing and power model timing and power model timing and power model
Verilog/	tsmc13_neg.v	Verilog simulation model
VHDL/	tsmc13.vhd	VHDL simulation model



## Design Report Form

隊號(Team number):		帳號(login id):	
<b>RTL category</b>			
<i>Design Stage</i>	<i>Description</i>	<i>File Name</i>	
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)		
<b>Gate-Level category</b>			
<i>Design Stage</i>	<i>Description</i>	<i>File Name</i>	
Pre-layout Gate-level Simulation	Gate-Level 檔案名稱 (Gate-Level Netlist file name)		
	Pre-layout sdf 檔案名稱		
	DC 資料庫檔案名稱(ddc)		
	Gate-Level simulation, 所使用最 小的 CYCLE Time	(                      ) ns	
	Design Latency		
<b>Physical category</b>			
<i>Design Stage</i>	<i>Description</i>	<i>File Name or Value</i>	
P&R (Optional)	使用之 P&R Tool (請填入 Astro 或 ICC 或 SOC Encounter)		
	設計資料庫檔案名稱(Library name)		
	佈局檔檔案名稱(GDSII file name)		
	佈局面積(layout area)	(                      ) um X (                      ) um	
	佈局座標點	左下角座標點(Lower-Left Coordinate) : XLB =                      YLB = 右上角座標點(Upper-Right Coordinate): XRT =                      YRT =	
	DRC report file		
	LVS report file		
	Post-layout Gate-level Simulation (Optional)	Post-layout Gate-Level 檔案名稱	
Post-layout sdf 檔案名稱			
Post-layout simulation, 所使用 最小的 CYCLE Time		(                      ) ns	
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			

