2013 University/College IC Design Contest

Cell-Based IC Design Category for Graduate Level

Sample Adaptive Offset Filter

1.問題描述

請完成一 Sample Adaptive Offset Filter (**後文以 SAO 表示**)的電路設計，輸入為一已分成多個 LCUs(如圖一所示)的影像，SAO 對每一 LCU 各自進行獨立運算，運算後的結果，請儲存於該電路內部的記憶體 sram_16384x8，最後整張影像訊號處理完後，將 finish 訊號拉為 High，系統會自動進行比對整張影像資料的正確性。有關 LCU 的定義與 SAO 詳細的運算方式，將描述於後。

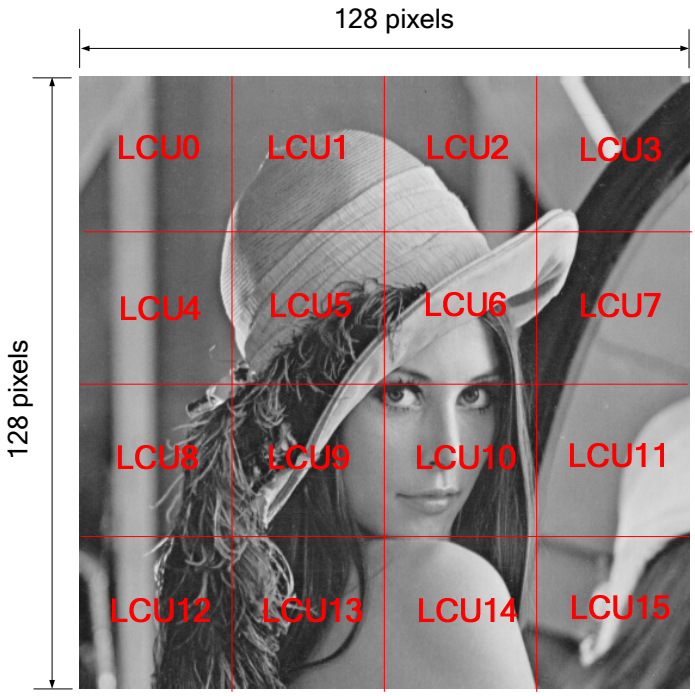
本電路各輸入輸出信號的功能說明，請參考表 1。每個參賽隊伍必須根據下一節所給的設計規格及附錄 A 中的測試樣本完成設計驗證。

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/icc2013/gcb/icc2013cb.tar*** ，請執行以下指令取得測試樣本：

tar xvf /usr/cad/icc2013/gcb/icc2013cb.tar

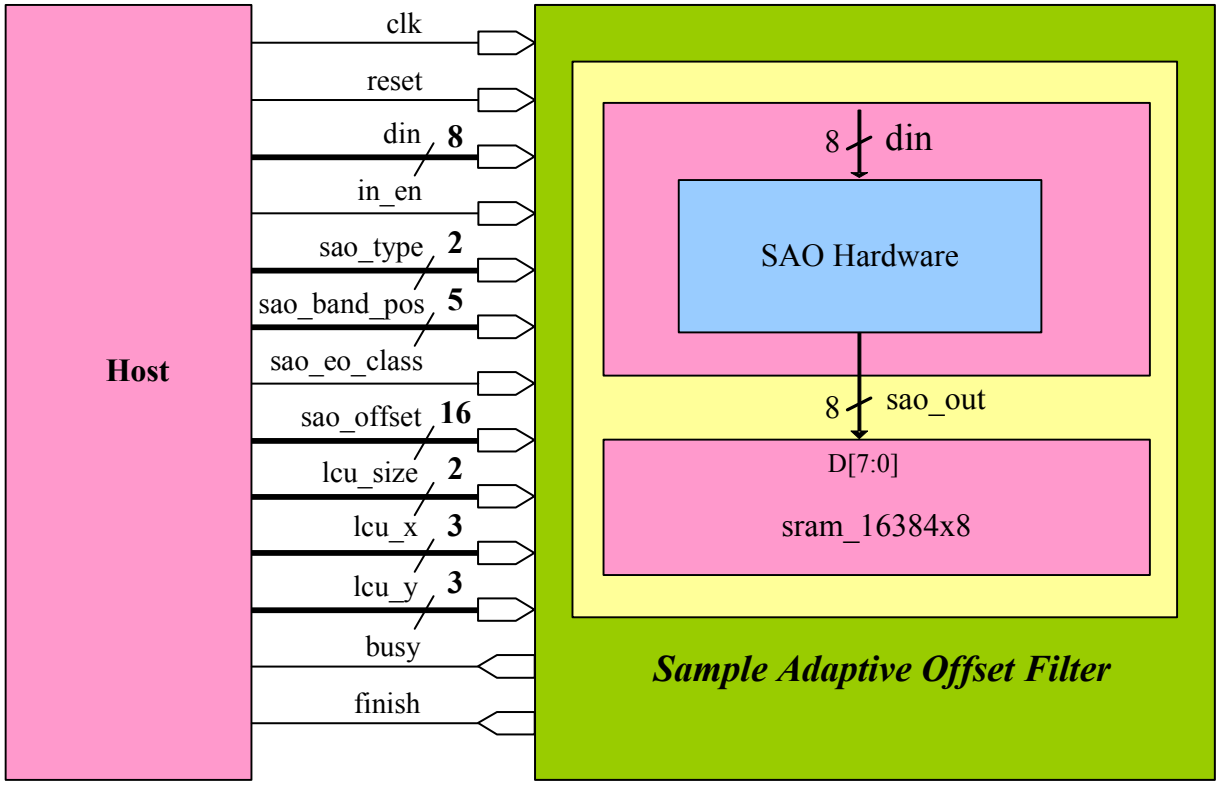
軟體環境及設計資料庫說明請參考附錄 F 與附錄 G。



圖一、將一張影像切割成多個 LCUs 之範例

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。
din	I	8	SAO 資料輸入的匯流排。Host 端會透過此匯流排將整張完整影像的訊號進行輸入。每一個週期僅能輸入一個 Pixel 值，且已輸入過的 Pixel 值將無法再重複輸入。註：輸入順序請參照 2.3.2。
in_en	I	1	資料輸入致能控制訊號。當 Host 偵測到 busy 訊號為 Low，din 便開始輸入訊號，此時 in_en 開始為 High，din 訊號輸入過程中，無論 busy 訊號為何，in_en 一直維持為 High，直到整張影像輸入完畢，in_en 才會為 Low。當 Host 端所有資料送完後，該訊號到模擬結束前將永遠維持為 Low。

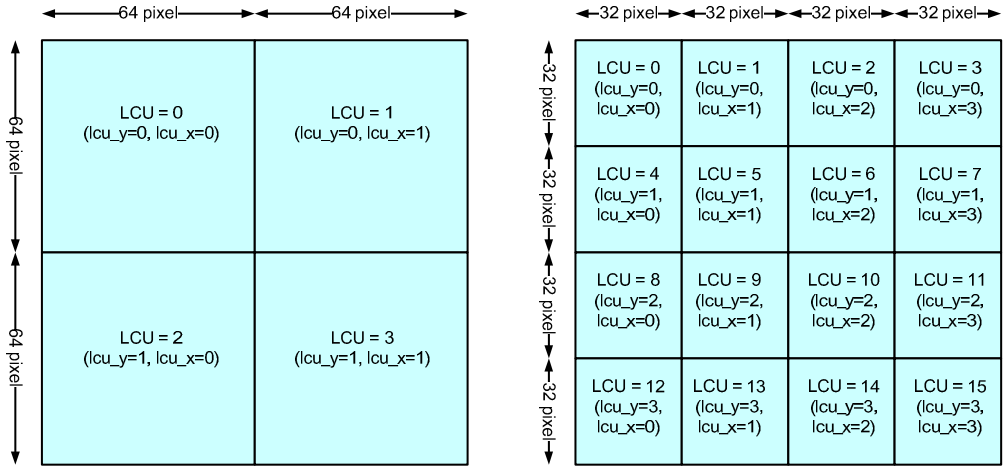
sao_type	I	2	指定目前輸入 LCU 之 SAO 運算型式，包括： (1)Type=2'd0，進行 SAO-OFF 運算。 (2)Type=2'd1，進行 BO 運算。 (3)Type=2'd2，進行 EO 運算。
sao_band_pos	I	5	指定目前輸入 LCU 進行 BO 運算之 First Band index 值，該值範圍為：0 ~ 28。sao_band_pos 僅有當 sao_type=2'd1 時有效；其餘 sao_type 不使用 sao_band_pos。
sao_eo_class	I	1	指定目前輸入 LCU 進行 EO 運算之兩種類別，包括： (1) 1'b0：水平方向濾波(Horizontal Edge Filter)。 (2) 1'b1：垂直方向濾波(Vertical Edge Filter)。 sao_eo_class 僅有當 sao_type=2'd2 時有效；其餘 sao_type 不使用 sao_eo_class。
sao_offset	I	16	指定目前輸入 LCU 進行 BO 或 EO 運算之 Offset 值。 4 Offset： offset_0= sao_offset[15:12], offset_1= sao_offset[11:8] offset_2= sao_offset[7:4], offset_3=sao_offset[3:0] (1) 若當作 BO 運算之 Offset (offset range = -7 ~ +7) offset_0：for 1st band of 4 continuous bands offset_1：for 2nd band of 4 continuous bands offset_2：for 3rd band of 4 continuous bands offset_3：for 4th band of 4 continuous bands (2) 若當作 EO 運算之 Offset offset_0：for category 1 (offset range = 0~+7) offset_1：for category 2 (offset range = 0~+7) offset_2：for category 3 (offset range = -7~0) offset_3：for category 4 (offset range = -7~0) sao_offset 僅有當 sao_type=2'd1 或 2'd2 時有效；其餘 sao_type 不使用 sao_offset。
lcu_size	I	2	指定整張影像訊號切割成多個 LCU 之尺寸大小。 (1) lcu_size = 0，每一塊 LCU 大小為 16x16 個 pixels。 (2) lcu_size = 1，每一塊 LCU 大小為 32x32 個 pixels。 (3) lcu_size = 2，每一塊 LCU 大小為 64x64 個 pixels。
lcu_x	I	3	目前輸入的 LCU 是座落於一張影像的哪個水平方位。 例如：一張影像 128x128 pixels，一個 LCU 為 16x16，因此水平方向最多有 128/16=8 個 LCU。 => lcu_x 合理範圍為： 0 ~ 7。
lcu_y	I	3	目前輸入的 LCU 是座落於一張影像的哪個垂直方位。 例如：一張影像 128x128 pixels，一個 LCU 為 16x16，因此垂直方向最多有 128/16=8 個 LCU。 => lcu_y 合理範圍為： 0 ~ 7。

busy	O	1	SAO 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，暫停 din 資料的輸入；反之，當為 Low 時，表示告知 Host 端可繼續由 din 輸入資料。
finish	O	1	SAO 運算完畢之通知訊號。當所有的 LCU 經過個別運算完畢後，需將 finish 訊號拉為 High，以通知 Host 端，開始進行所有影像訊號之比對。

2.3 系統描述

2.3.1 SAO 系統架構

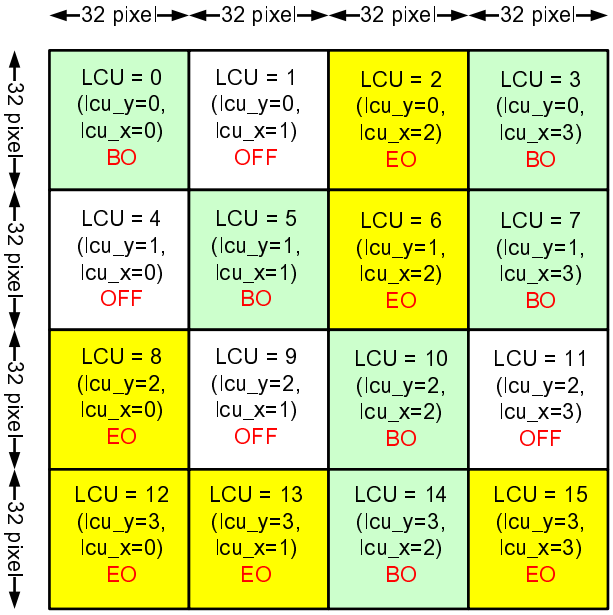
SAO 為新一代高效率視訊編碼 (High Efficiency Video Coding，簡稱 HEVC)技術，在 Coding 之前，會先將一張影像切割成多個 LCUs(Large Coding Unit)，本題 SAO 輸入影像大小固定為 128x128 pixels，每個 pixel 為 8bit 灰階，LCU 大小依據 lcu_size 可分成 16x16、32x32、64x64 三種尺寸。如圖三(b)所示，當 LCU Size 為 32x32，水平方向可切割出 128/32=4 個 LCUs，垂直方向也可切割出 4 個 LCUs，因此總共可切割出 16 個 LCUs。在同一張影像的每一個 LCU 都是相同大小。



(a) lcu_size=2　　(b) lcu_size=1

圖三、LCU 不同 Size 之切割範例(a)LCU Size：64x64，(b)LCU Size：32x32

切割後的每一塊 LCU，都會指定一項 SAO 運算的任務，如圖四所示，參賽者需要依據每個 LCU 所指定的運算任務，個別完成運算後再放入 SAO 電路內部的記憶體 sram_16384x8，即完成整個電路設計。SAO 運算有三種，包括(1)OFF、(2)BO、(3)EO，這三種運算方式，將詳細介紹如下。



圖四、指定每個 LCU(32x32)一個 SAO 運算的任務之範例

2.3.1.1 SAO-OFF (OFF)運算方式

當 sao_type 輸入 2'd0，表示該 LCU 要進行 OFF 運算。所謂 OFF，就是整個 LCU 內的所有 Pixels 數值，保持不變，直接寫入內部記憶體 sram_16384x8。

2.3.1.2 Band Offset (BO)運算方式

當 sao_type 輸入 2'd1，表示該 LCU 要進行 BO 運算。BO 運算對單一 pixel 做各別處理，一個 8bit pixel 其數值範圍為 0~255，固定分成 32Bands，即每個 Band 有包含 256/32=8 個不同的 Pixel 值，如圖五所示，Band 0 為 Pixel 數值 0~7，Band 1 為 Pixel 數值 8~15，……，Band 31 為 Pixel 數值 248~255。

進行 BO 運算時，Host 端會先透過 sao_band_pos 訊號輸入待處理的 LCU 其 first band index 的數值，例如 sao_band_pos=8，其往後的四個連續 Bands(8、9、10、11)就是要做 BO 運算的區間，這區間以外的 Pixel 值不需做任何改變，落在此區間的 Pixel 就要依所在的區間，根據 sao_offset 所指定的 offset 數值相加後，取代原本的 Pixel 值。

sao_offset 為 4 個 band offset 的合併， 請參考表 1 - sao_offset 說明。

sao_offset

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
← offset_0 →				← offset_1 →				← offset_2 →				← offset_3 →			

例如：落在 Band 8 範圍的 Pixel，就要加上 offset_0 的數值，取代原本的 Pixel 數值；落在 Band 9 範圍的 Pixel，就要加上 offset_1 的數值，取代原本的 Pixel 數值；……；落在 Band 11 範圍的 Pixel，就要加上 offset_3 的數值，取代原本的 Pixel 數值。有關 BO 細項運算可參考圖五之範例。

註 1：運算後的數值，Pixel 數值要永遠保持在 0~255 之間，亦即加上 offset 值後，超過 255，就用 255 表示，加上 offset 值後，低於 0，就用 0 表示。

註 2：由於 sao_band_pos 是指 first band index of 4 continuous bands，而 band 最大為 band31，因此 sao_band_pos 的有效範圍為 0~28。

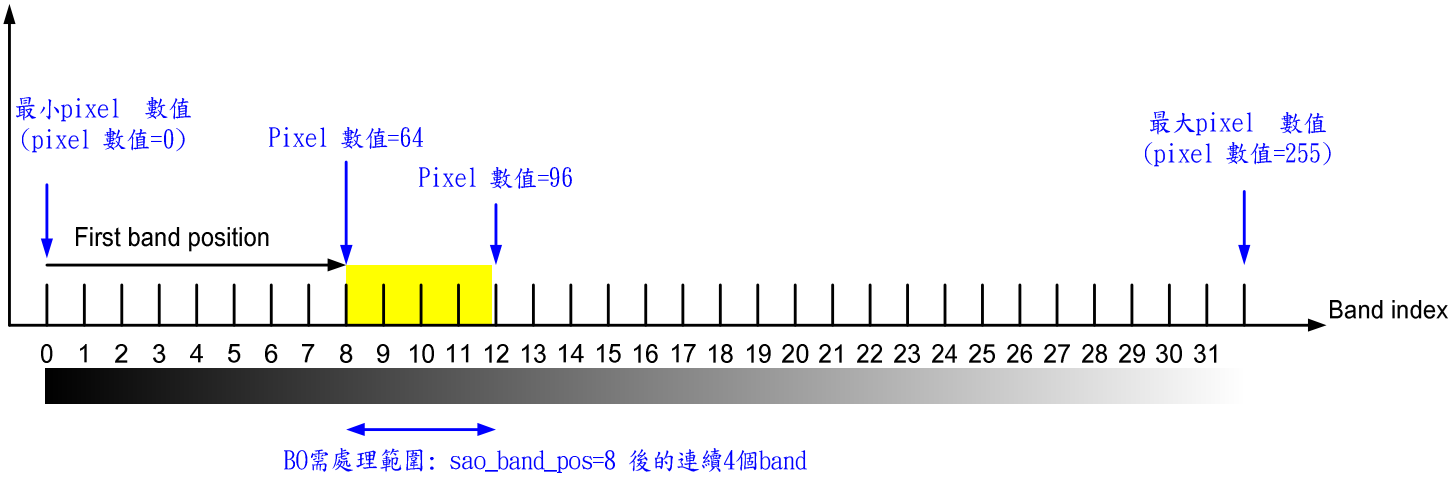
For exmaple,

sao_type=2'd1 => *BO Operation*

sao_offset=16'h73DA => *offset value = {+7, +3, -3, -6}*

sao_band_pos=5'd8 => *1st band of 4 continuous bands is 8*

sao_band_pos = 8	Pixel Intensity
1st band range (band 8)	64 ~ 71
2nd band range (band 9)	72 ~ 79
3rd band range (band 10)	80 ~ 87
4th band range (band 11)	88 ~ 95

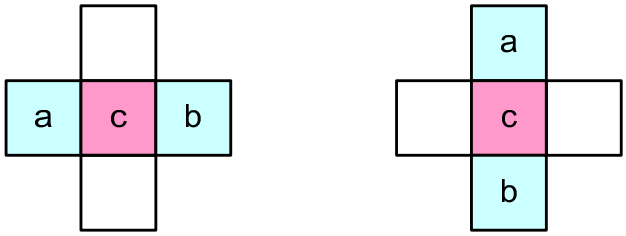


BO Operation:

Assume Input Pixel	Band Range	New Pixel with BO Operation
70	1st	70+offset_0=70+sao_offset[15:12] =70+7=77
95	4th	95+offset_3=95+sao_offset[3:0] =95-6=89
72	2nd	72+offset_1=72+sao_offset[11:8] =72+3=75
84	3rd	84+offset_2=84+sao_offset[7:4] =84-3=81
98	out of range	98

圖五、BO 運算的範例

2.3.1.3 Edge Offset (EO)運算方式



sao_eo_class=1'b0
(horizontal)

sao_eo_class=1'b1
(vertical)

category	Condition
1	c < a && c < b
2	(c<a && c==b) || (c<b && c==a)
3	(c>a && c==b) || (c>b && c==a)
4	c > a && c > b
0	None of the above

Offset value range : 0~+7
(must be positive)

Offset value range : -7~0
(must be negative)

圖六、EO 運算法則

當 sao_type 輸入 2'd2，表示該 LCU 要進行 EO 運算。EO 運算有分成(1)sao_eo_class=1'b0，horizontal edge filter、(2)sao_eo_class=1'b1，vertical edge filter 兩種方法，兩種方法都會依據圖六所列之 Condition，判斷 c 與相鄰 a、b 兩個 Pixels 之間的關係做 category 分類，若屬於 category 1，便將原始的 pixel 值+offset_0 後取代原始的 Pixel 值，依此類推，若屬於 category 4，便將原始的 pixel 值+offset_3 後取代原始的 Pixel 值，假若圖六之四個 Condition 都沒滿足，即為 category 0，表示原始的 Pixel 值不做任何改變。有關 EO 細項運算可參考圖七之範例。

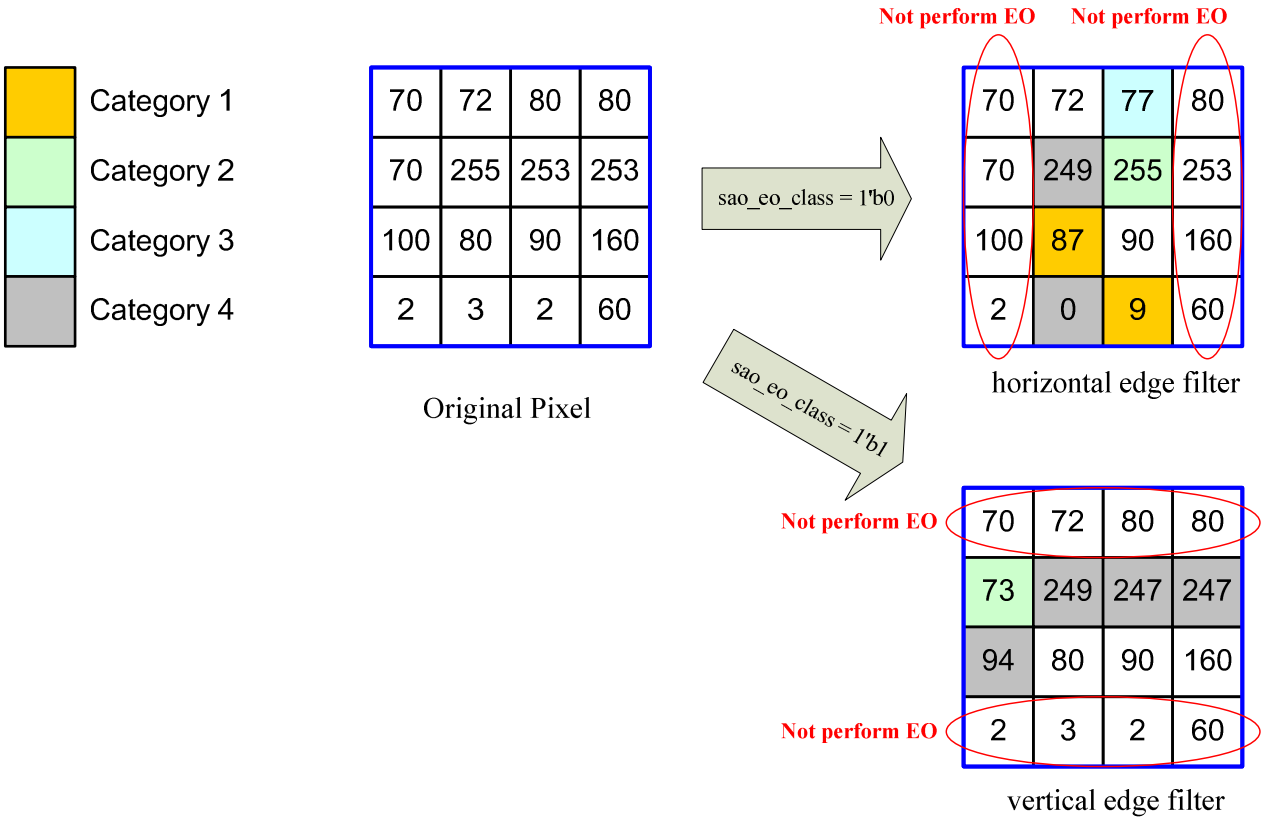
註 1：圖六的 a、b、c 之 Pixel 數值，僅可使用原始影像 Pixel 值，請勿使用運算後之 Pixel 值。
註 2：每個 LCU 作 EO 運算時，只會限定水平或垂直之其中一種 EO Class。
註 3：LCU 邊緣的 Pixels，不作計算，所以當執行 horizontal edge filter 時，最左及最右邊的 LCU Pixel 值保持不變，當執行 vertical edge filter 時，最上及最下面的 LCU Pixel 值保持不變
註 4：運算後的數值，Pixel Intensity 要永遠保持在 0~255 之間，亦即加上 offset 值後，超過 255，就用 255 表示，加上 offset 值後，低於 0，就用 0 表示。

For exmaple, LCU size: 4x4

sao_type=2'd2 => *EO Operation*

sao_offset=16'h73DA => *offset value = {+7, +3, -3, -6}*

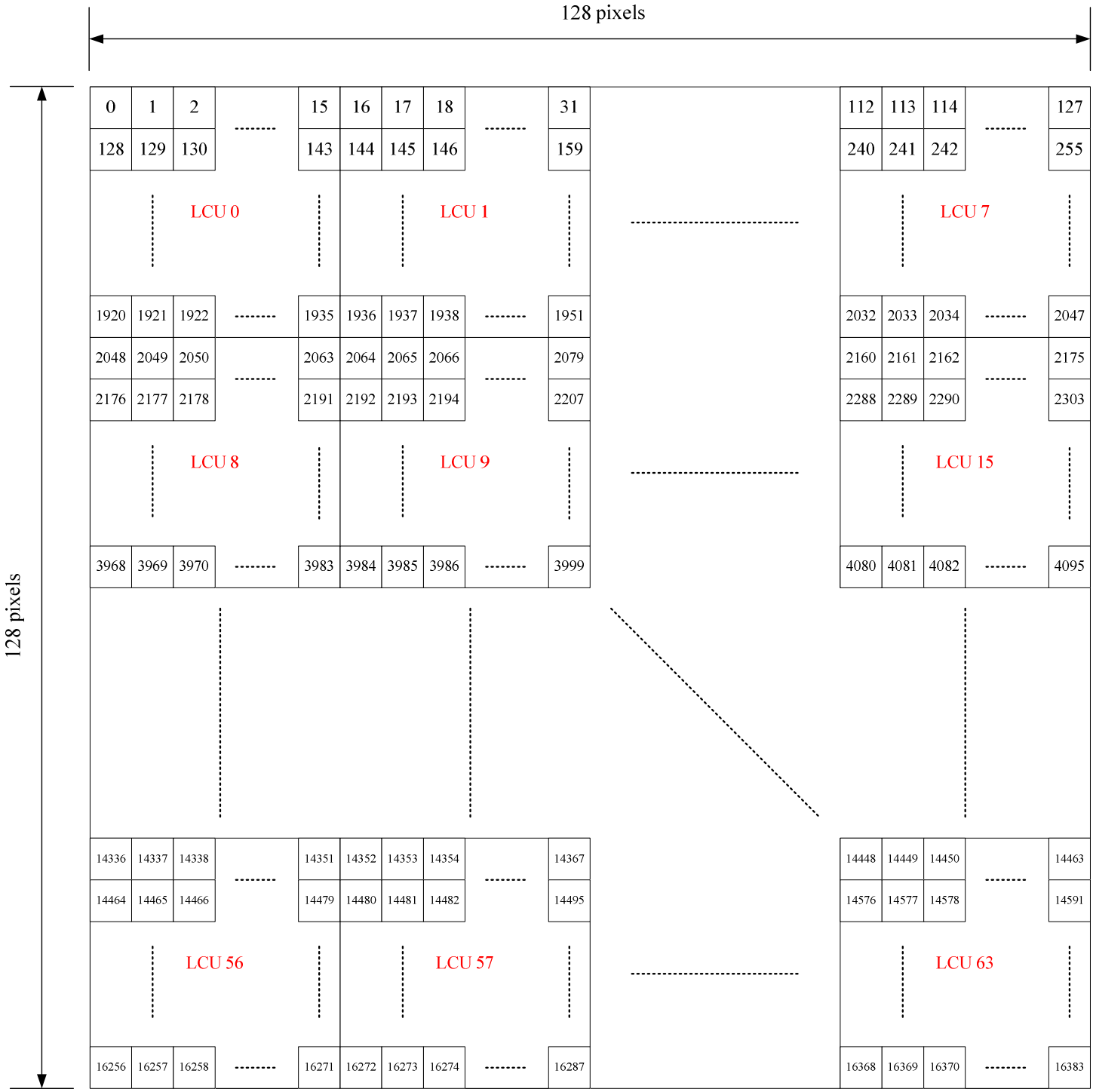
Begin to perform EO operation:



圖七、EO 運算範例

2.3.2 SAO 運算之輸入方式

一張 8bits 128x128 影像如圖一所示，若要將該影像儲存在 1D 的記憶體，其儲存方式如圖八所示，注意圖八的編號為記憶體位址值，因此一個 16x16 的 LCU 其儲存的影響 Pixel 值並非為連續記憶體位址儲存而成，而是例如 LCU0 是由 SRAM Address 0~15、128~143、…、1920~1935 共計 16x16=256 個位址，256 個 Pixels 值所構成，為簡化其複雜度，本題影像的輸入訊號 din 送入 SAO 電路的順序為 Raster Scan(如圖八之 LCU size 16x16 為例，順序為 LCU 0 -> LCU 63，其中 LCU 0 的順序為 0~15、128~143、…、1920~1935，其餘 LCU 1~63 也依此順序類推)，另外，每當一個 LCU 資料 din 開始輸入時，sao 相關參數(sao_type、sao_band_pos、sao_eo_class、sao_offset)及 LCU 相關參數(lcu_size、lcu_x、lcu_y)，也將跟著進入 SAO 電路，四個 sao 相關參數不見得每一種 sao_type 運算都會使用到。



圖八、將一張 128x128 影像訊號儲存於 16KB SRAM 記憶體之範例

註 1：已輸入至 sao 電路的訊號 din、sao、lcu 相關參數，無法再輸入第二次，請注意!倘若輸入的訊號暫時不想接收，可以用 busy 訊號拉為 High，即可暫停資料輸入，待處理完後，再將 busy 訊號拉維 Low，即可繼續輸入剩餘的資料。

註 2：本題 Image Size 永遠固定為 128x128 Pixels，但 LCU Size 依舊有 16x16、32x32、64x64 三種可能。

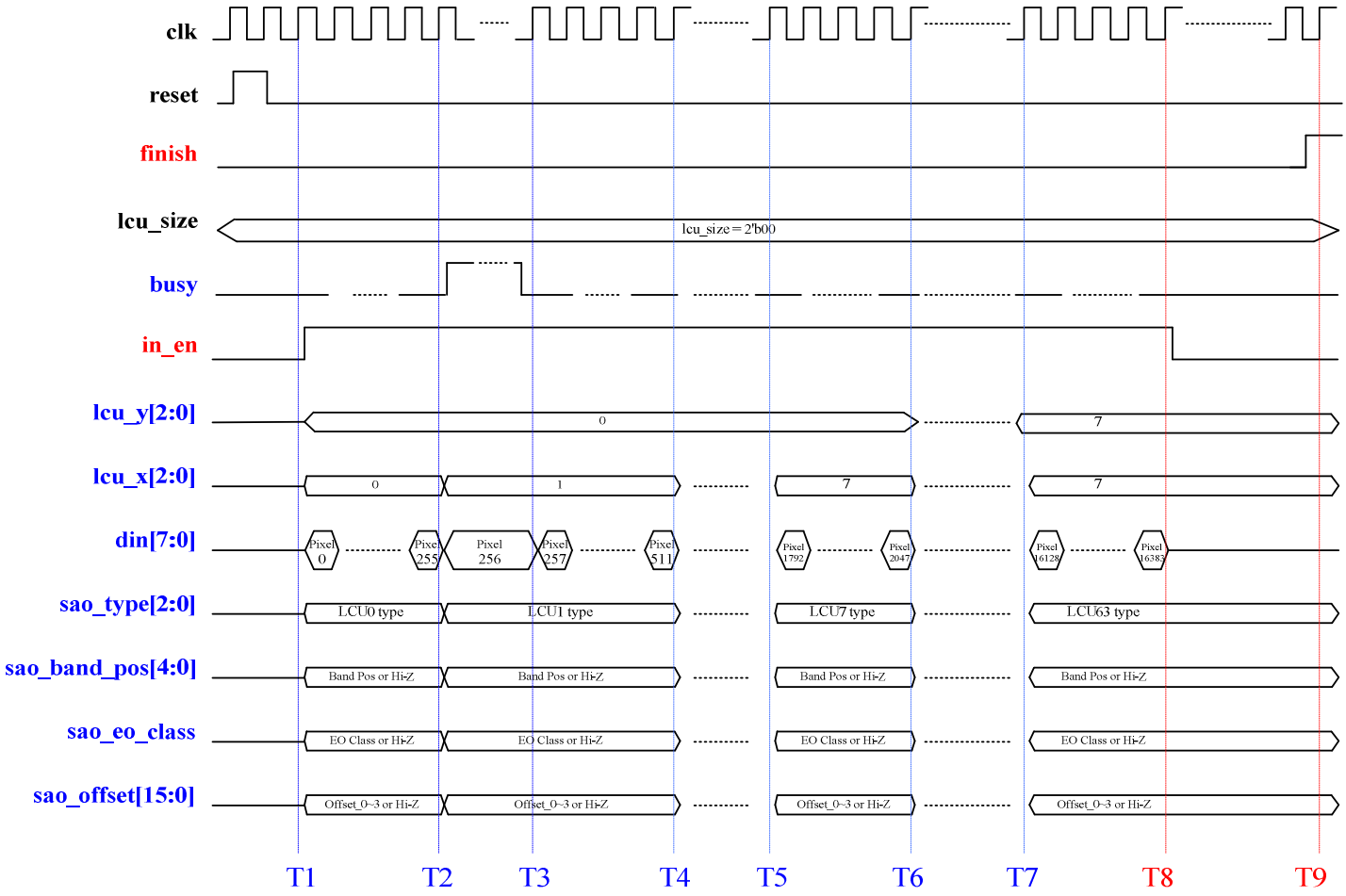
2.3.3 SAO 運算之輸出方式

本題之 SAO 運算完結果不須輸出，請參賽者自己找適當時間將運算完結果寫入至內部記憶體 sram_16384x8，其算完後擺放的位址，以 LCU Size 16x16 為例，會與圖八記憶體擺放順序相同，意即 LCU 0 運算完的結果，不會連續儲存 256 個位址，而是要自行儲存在 SRAM Address 0~15、128~143、…、1920~1935 共計 256 個位址裡，其餘 LCU 也依此類推，請注意！最後所有的 LCU 都運算完後，請務必將 finish 訊號拉為 High，以通知 Host 端開始進行整個 16KB SRAM 的資料比對。

2.4 電路時序規格

2.4.1 SAO 電路時序規格

本時序圖以 LCU Size：16x16 為例，其餘 Size 依此類推。



圖九、SAO 電路時序圖

1. T1 時間點，reset 一個 Cycle 的時間，SAO 電路初始化結束，Host 端在 T1 時間點判斷到 busy 訊號為 Low，因此開始輸入資料，in_en 拉為 High，din、lcu_y、lcu_x、din、sao_type、sao_band_pos、sao_eo_class、sao_offset 一起送出第一筆資料，開始運算第一個 LCU 0。

2. T2 時間點，經過 Host 端送出 256 筆 Pixels，lcu_x 便加 1，在 T2~T3 期間，LCU0 未必已完成計算，因此過程中可能會將 busy 訊號拉為 High，輸入的資料就會有如 T2~T3 這段時間看到的一樣，例如 Pixel 256 有輸入但是 busy 為 High，因此暫時維持 din 資料一直是 Pixel256，直到 T3，clk 正緣來發現 busy 是 Low，才會再送下一筆資料 Pixel 257。往後的 busy 訊號的動作行為，也是依此類推，就不再贅述了。

3. T4 時間點， LCU1 的 256 筆 Pixels 全數送完後，過程中或送完後還會有其餘運算要作，請參賽者隨時使用 busy 訊號作好控制，因為送進來過的 Pixel 訊號就不會再送一次了。完成 LCU 1 運算後，lcu_x 又會再加 1，就這樣類似動作一直加到 lcu_x=7 後，如圖之 T1~T6，lcu_y 便會加 1，最後 lcu_y 加到等於 7，lcu_x 也等於 7，就會像 T7~T8，便完成整張影像的輸入。

4. T8 時間點，資料全數輸入完成，in_en 拉為 Low，表示不再有新資料輸入，接下來到 SAO 運算完 finish 之前，請趕緊把剩餘資料算完後，於 T9 時間點發出 finish 為 High 後，便告知 Host 端開始進行 SRAM 之資料比對，比對完後整個模擬立即結束。

註 1：比對 SRAM 資料的期間，不會浪費任何 Simulation Time 的。

2.4.2 SARM 記憶體規格與時序規格

製作 SAO 電路時，參賽者可能會用到記憶體當 Buffer，主辦單位在此提供三種記憶體包括：256x8、4096x8、16384x8 等三種記憶體，請自行決定哪個記憶體最適合自己的設計，請自行採用。另外，由於運算完結果會放置 sao 電路內部電路的 sram_16384x8，參賽者請務必記得將 sram_16384x8 記憶體呼叫進來，進行合成與 APR，以完成整個設計。

有關三種 SRAM 記憶體細節規格與記憶體之時序圖，詳如記憶體 PDF 檔案中。

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 = 238649 NS 做評分。

```
----------------------------------------------------

Congratulations! All data have been generated successfully!

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

Simulation complete via $finish(1) at time 238649 NS + 0
```

註 1：本題有三種 LCU Size 的模擬，Simulation Time 請以三者之最大值為準。

✧ **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} { 556.020 552.430}

=> Area = 556.020x552.430 =307162.1286 um^2

2. Encounter Report Area 範例:

encounter > analyzeFloorplan

********** Analyze Floorplan **********

Die Area(um^2)	: 292970.92
Core Area(um^2)	: 279813.18
Average utilization	: 100.000%
Number of instance(s)	: 4605
Number of Macro(s)	: 2
Number of IO Pin(s)	: 45
Number of Power Domain(s)	: 0

********** Estimation Results **********

=> Area = 292970.92 um2

請注意：指令 analyzeFloorplan 會破壞已完成 routing 的結果，**執行過此指令後千萬不能再存檔。**

設計完成程度三種等級，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 Golden Pattern 比對完全正確。

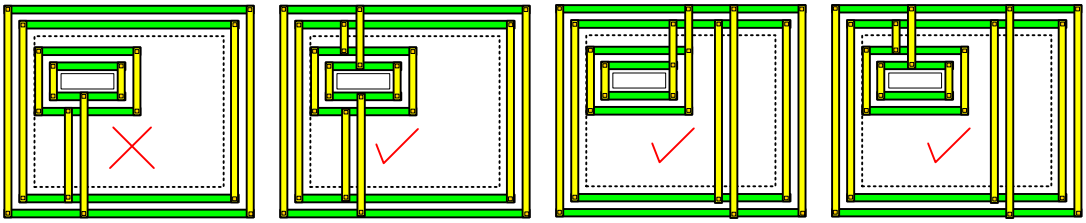
b、 完成 Synthesis，且 Gate-Level Pre-layout Simulation 結果正確。

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. 內建的所有記憶體 SRAM，其 VDD、VSS Pin 務必要連接至 Core Power Ring，寬度請各設定為 2um。
v. 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)
vi. 務必要加 Power Rail (follow pin)。
vii. Core Filler 務必要加。
viii. APR 後之 GDSII 檔案務必產生。
ix. 完成 APR，DRC/LVS 完全無誤(見附錄 C 說明)。

註: Power Stripe 指的是直接穿過 core area 的 power line，見下圖



等級 A 之評分方法：

Score = Time x Area

例如：

在前一頁範例中(Encounter)，Score = Time x Area = 238649 x 292970.92 = 69917217087

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 = Time x Area 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 = Time x Area

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 RTL Simulation，結果影像 pixel 錯誤總數量越少者，分數越高。

Score = Total Pixel Errors of RTL Simulation

註:

1. Score 越小者，同級名次越好!
2. 等級 D，視合成 & APR 為 Fail，Area、Timing 將不予考慮之。
3. 等級 D，只以 RTL Simlations 正確率為主。

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 2、設計檔案說明

檔名	說明
SAO.v	參賽者所使用的設計檔，已包含系統輸/出入埠之宣告及放置 SAO 運算結果會用到的 SRAM。
testfixture1.v testfixture2.v testfixture3.v	三個 Test Bench 檔案。每個 Test Bench 已分別加入三種 LCU Size：16x16、32x32、64x64 及三種 LCU Size 運算完後需比對的 golden 檔案。
image_16x16.dat image_32x32.dat image_64x64.dat	三組影像 din 的輸入訊號來源。這三組影像雖然皆為同一張 Lena 影像，採 RasterScan 順序輸入，但由於 LCU Size 不同，所以有三種輸入順序，請參賽者模擬時請勿用錯檔案。
lcu_16x16.dat lcu_32x32.dat lcu_64x64.dat	三種 LCU Size 其個別的 sao_type、sao_band_pos、sao_eo_class、sao_offset 之輸入訊號來源。例如：LCU Size=16x16，對於 128x128 影像就會有 64 個 LCUs，因此在 lcu_16x16.dat 檔案中，會有 64 個 LCUs 的 sao 相關資訊。該 Type 不需要用到的訊號，會以 High-Z 訊號表示。
golden_16x16.dat golden_32x32.dat golden_64x64.dat	三組 Golden Pattern 對應到三種 LCU Size 經過 SAO 運算後之結果，從檔名可以看出是哪一種 LCU Size 運算完的 Golden 結果。
.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。

SAO_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，其餘環境參數請勿更改。
SAO_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，其餘環境參數請勿更改。
sram_16384x8_slow_syn.db	SRAM timing library (For DC and ICC) 注意:無論 DC 合成或用 ICC 做 APR,只需使用 worst case library，例如： slow.db、sram_16384x8_slow_syn.db 。
sram_16384x8_slow_syn.lib	SRAM timing library(For SOCE)。 注意：用 Encounter 做 APR，只需使用 worst case library 例如：slow.db、sram_16384x8_slow_syn.db。
sram_16384x8	SRAM Frame View，ICC APR 專用。
sram_16384x8.vclef	SRAM LEF，Encounter APR 專用。
sram_16384x8.gds	SRAM GDSII，作為 Stream Out GDSII 之用。

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

```
module SAO (clk, reset, in_en, din, sao_type, sao_band_pos, sao_eo_class, sao_offset,
                lcu_x, lcu_y, lcu_size, busy, finish);
input     clk;
input     reset;
input     in_en;
input     [7:0]    din;
input     [1:0]    sao_type;
input     [4:0]    sao_band_pos;
input              sao_eo_class;
input     [15:0] sao_offset;
input     [2:0]    lcu_x;
input     [2:0]    lcu_y;
input     [1:0]    lcu_size;
output    busy;
output    finish;

   sram_16384x8    golden_sram(.Q( ), .CLK( ), .CEN( ), .WEN( ), .A( ), .D( ));
endmodule
```

2. 為因應三種 LCU Size，主辦單位在此提供三種 Test Bench：testfixture1.v, testfixture2.v, testfixture3.v 分別對應到 LCU Size：16x16、32x32、64x64，其相關會用到的檔案都已加入，如下：

例如：

第一種 LCU Size：16x16，請使用 testfixture1.v 作模擬：

```
`define IMAGE_PAT "./image_16x16.dat"
`define LCU_PAT   "./lcu_16x16.dat"
`define EXP       "./golden_16x16.dat"
```

註：參賽者無須作修改，只需注意 image_16x16.dat、lcu_16x16.dat、golden_16x16.dat 的檔案位置即可，預設路徑為"目前目錄"。

3. 主辦單位所提供的三個 Test Bench 檔案，有多增加幾行特別用途的敘述如下：

```
`define  End_CYCLE  1000000
`define  SDFFILE    "./SAO_syn.sdf"
`ifdef  SDF
    initial  $sdf_annotate(`SDFFILE, SAO);
`endif
```

註：

1. End_CYCLE 預設 100 萬個 Cycles，其目的可以防止參賽者因電路有錯，模擬陷入無窮回圈之境，倘若參賽者確定模擬需超過 100 萬個 Cycles 以上，可自行再加大此 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  SAO_syn.v  sram_16384x8.v
  –v  tsmc13_neg.v  +define+SDF  +access+rw
```

註 1：若參賽者有用到其餘 SRAM 作 Buffer，請模擬時自行將相對的.v 檔案一同加入作模擬。

註 2：使用 Encounter 作 APR 之參賽者，APR 後的模擬務必加上+ncmaxdelays 參數，請注意！

```
ncsim> run
Send Data Over!
Begin to Check SRAM Data....
--------------------------------------------------
--------------------------------------------------

Congratulations! All data have been generated successfully!

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

Simulation complete via $finish(1) at time 238649 NS + 0
./testfixture1.v:192        #(`CYCLE/2); $finish;
ncsim> exit
```

附錄 B 測試樣本

本題之測試樣本皆使用如圖一之 Lena 影像檔作輸入，但由於 LCU Size 不同，因此可分成 image_16x16、image_32x32、image_64x64 三組資料檔及 lcu_16x16、lcu_32x32、lcu_64x64 三組 sao 相關資訊。由於考量本題實現上比對資料正確與否不容易，因此主辦單位針對 LCU Size：16x16 的版本，提拱 LCU 0、LCU 1 的原始 Pixel 數值與 SAO 運算後的 Pixel 數值，以方便實現本電路。從 lcu_16x16.dat 檔案可以看出，LCU 0 是作 BO 運算，offset ={1, 7, -3, -4}，LCU 0 的運算後結果可參考圖 10。LCU 1 是作 EO 之 vertical edge filter 運算，offset ={2, 6, -1, -6}，LCU 1 的運算後結果可參考圖 11。

AA	99	99	99	AA	99	99	99	AA	99	AA	AA	AA	99	99	77
AA	99	99	99	99	99	99	99	99	99	BB	99	BB	99	99	66
99	99	99	99	99	99	99	99	99	99	AA	99	AA	99	88	66
AA	99	99	99	AA	99	99	99	AA	99	BB	99	BB	99	99	66
99	99	99	99	AA	99	99	99	AA	99	AA	99	AA	99	77	66
99	99	99	99	99	88	99	99	AA	99	AA	99	AA	99	99	66
99	99	99	99	99	99	99	99	AA	99	99	99	99	99	88	66
AA	99	99	99	AA	99	AA	99	BB	99	99	99	AA	99	99	66
AA	99	99	99	AA	99	AA	AA	AA	99	99	99	AA	99	88	66
AA	99	99	99	AA	99	BB	99	AA	99	99	99	AA	99	99	66
99	99	AA	99	AA	99	AA	99	99	99	99	99	99	99	88	66
AA	99	AA	99	BB	99	99	99	99	99	AA	99	BB	99	99	66
AA	99	99	BB	AA	99	88	88	88	99	99	99	AA	99	88	66
AA	99	BB	99	AA	88	88	88	99	99	AA	99	AA	99	99	66
99	99	AA	99	99	88	77	88	99	99	AA	99	AA	99	88	66
AA	99	BB	99	99	66	66	88	AA	99	BB	99	BB	99	99	66

(a)

AA	99	99	99	AA	99	99	99	AA	99	AA	AA	AA	99	99	78
AA	99	99	99	99	99	99	99	99	99	BB	99	BB	99	99	66
99	99	99	99	99	99	99	99	99	99	AA	99	AA	99	84	66
AA	99	99	99	AA	99	99	99	AA	99	BB	99	BB	99	99	66
99	99	99	99	AA	99	99	99	AA	99	AA	99	AA	99	78	66
99	99	99	99	99	84	99	99	AA	99	AA	99	AA	99	99	66
99	99	99	99	99	99	99	99	AA	99	99	99	99	99	84	66
AA	99	99	99	AA	99	AA	99	BB	99	99	99	AA	99	99	66
AA	99	99	99	AA	99	AA	AA	AA	99	99	99	AA	99	84	66
AA	99	99	99	AA	99	BB	99	AA	99	99	99	AA	99	99	66
99	99	AA	99	AA	99	AA	99	99	99	99	99	99	99	84	66
AA	99	AA	99	BB	99	99	99	99	99	AA	99	BB	99	99	66
AA	99	99	BB	AA	99	84	84	84	99	99	99	AA	99	84	66
AA	99	BB	99	AA	84	84	84	99	99	AA	99	AA	99	99	66
99	99	AA	99	99	84	78	84	99	99	AA	99	AA	99	84	66
AA	99	BB	99	99	66	66	84	AA	99	BB	99	BB	99	99	66

(b)

圖 10　LCU 0 (a) Original Pixels、(b) New Pixels with BO 之範例 (sao_band_pos=14)

55	66	66	66	77	66	66	66	77	66	66	77	77	66	77	88
66	44	66	66	66	66	66	66	77	66	66	66	88	77	88	66
55	55	55	66	66	66	66	66	66	66	77	66	77	88	77	66
66	55	66	66	66	66	66	66	66	66	77	66	88	66	88	77
55	55	66	55	77	66	66	66	66	66	66	66	77	66	77	88
66	55	66	66	66	55	66	66	66	66	77	66	88	77	88	66
44	55	55	66	66	66	66	66	66	66	66	66	77	88	77	77
55	55	66	66	77	66	66	66	66	66	77	66	88	66	88	77
55	55	55	55	77	66	66	55	55	66	66	66	77	66	77	88
66	44	66	66	66	55	66	66	66	55	66	66	88	77	88	66
44	55	55	66	66	66	66	66	66	66	66	66	77	77	77	66
55	55	66	66	77	66	66	66	66	66	77	66	88	66	88	77
55	55	66	55	66	66	66	66	66	66	66	66	77	66	77	88
55	44	66	66	66	55	66	66	66	66	77	66	88	77	88	66
44	55	55	55	66	66	55	66	66	66	77	66	77	77	77	66
55	55	66	55	66	66	66	55	66	66	66	66	88	66	88	66

(a)

55	66	66	66	77	66	66	66	77	66	66	77	77	66	77	88
60	46	65	66	6C	66	66	66	76	66	6C	6C	82	77	82	6C
57	54	57	66	66	66	66	66	6C	66	76	66	79	82	79	6C
60	55	65	65	6C	66	66	66	66	66	76	66	82	6C	82	77
57	55	66	57	71	65	66	66	66	66	68	66	79	6C	79	82
60	55	65	65	6C	57	66	66	66	66	71	66	82	77	82	68
46	55	57	66	6C	65	66	66	66	66	68	66	79	82	79	76
54	55	60	65	76	66	66	65	65	66	71	66	82	6C	82	7D
5B	54	57	57	76	65	66	57	57	65	6C	66	79	6C	79	82
60	46	60	65	6C	57	66	65	65	57	66	66	82	76	82	6C
46	54	57	66	6C	65	66	66	66	65	6C	66	79	76	79	6C
54	55	65	65	71	66	66	66	66	66	71	66	82	6C	82	77
55	54	66	57	6C	65	66	66	66	66	68	66	79	6C	79	82
54	46	65	60	66	57	65	66	66	66	76	66	82	76	82	6C
46	54	57	5B	66	65	57	65	66	66	76	66	79	76	79	66
55	55	66	55	66	66	66	55	66	66	66	66	88	66	88	66

(b)

圖 11　　LCU 1 (a) Original Pixels、(b) New Pixels with EO (Vertical) 之範例

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。

- 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	SAO_pr.sdf
Significant digits	2

   按 OK。

   對應指令： write_sdf -version 2.1 SAO_pr.sdf

   “ File > Export > Write Verilog… ”

   先按 Default

Output verilog file name	SAO_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	SAO_pr.v
All other options	Default value

   按 OK。

   “ Timing → Extract RC… ” ，按 OK。

“ Timing → Write SDF… ”

Ideal Clock	Disable
SDF Output File:	SAO_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

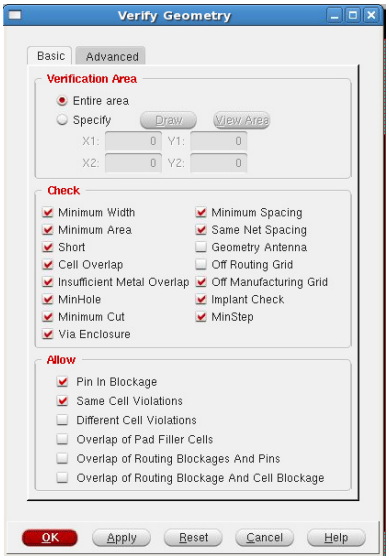
按 OK。

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：

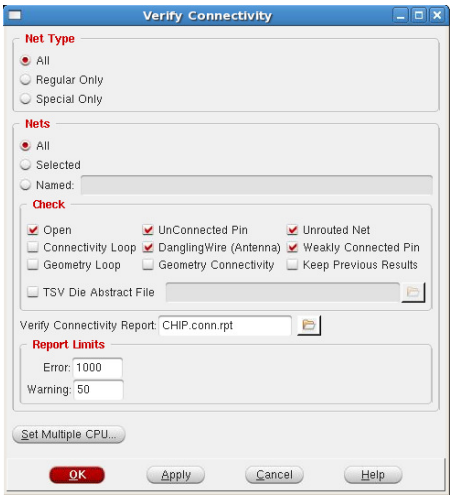
在 Encounter 視窗下點選

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 3

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 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 D 要求的檔案複製到 result 這個目錄。例如:
   > ***cp SAO.v ~/result/***
   > ***cp SAO_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. 主辦單位已將所有軟體環境設定於：/cad/icc2013/cshell/login.cshrc，參賽同學不需再做任何設定。

Vendor	Tool	Executable
Cadence	Virtuoso *1	icfb
	Composer	icfb
	NC-Verilog	ncverilog
	SOC Encounter	encounter
Synopsys	design vision	dv, dc_shell
	VCS	vcs
	IC compiler	icc_shell -gui
	Hspice	hspice
	Cosmos Scope *1	scope
	Spice explorer *1	wv
Mentor	Calibre *3	caliber
	ModelSim	vsim
Spring Soft	Laker *1	laker
	Verdi *1	verdi, nWave, nLint
Utility	vi	vi, vim, gvim
	gedit	gedit
	nedit	nedit
	pdf reader	acroread
	calculate	galculator, 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： **/usr/cad/icc2013/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
VHDL/	
tsmc13.vhd	VHDL simulation model
Phantom/	
tsmc13gfsg_fram.gds	Standard Cell GDSII file
Memory/	
sram_16384x8/	
sram_16384x8_slow_syn.db	DC ICC APR 可使用(worst-case)
sram_16384x8_slow_syn.lib	Encounter APR 可使用(worst-case)
sram_16384x8.v	Verilog simulation model
sram_16384x8.vclef	Encounter APR 可使用
sram_16384x8_ant.lef	Encounter 可使用(LEF for antenna)
sram_16384x8/	ICC 可使用(SRAM Fram View)
sram_16384x8.pdf	SRAM Spec Document
sram_16384x8.gds	SRAM GDSII file
sram_4096x8/	內容物與 sram_16384x8 描述相同
sram_256x8/	內容物與 sram_16384x8 描述相同

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 X ( ) um
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) Ex: 10ns	
	Post-layout Simulation Time (Simulation Time,ex: 238649 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		