期末報告

[1] 設計者姓名與連絡電話

學生姓名: 聶家任、吳宗翰

[2] 專題名稱

中文專題名稱：二元神經網路加速晶片

英文專題名稱：Binary Neural Network Accelerator Chip

[3] 全新設計或改版說明

此案件為設計者全新設計。

[4] 原理及架構說明

二元神經網絡(Binary Neural Network)是一種特殊的神經網絡架構，其特點是神經元的權重(Weight)和激活值(Activation)只取二元值，通常是-1 和 1，在電路中則是用 0 和 1 個別代表-1 和 1 兩種數值，故二元神經網路的硬體加速器(Hardware Accelerator)能在速度上、面積上、能量上都較其他的量化(Quantization)方式來得優異，且二元神經網路在一些簡單的情境下也能接近或達到其他量化方式所得到的最好結果，故二元神經網路晶片適合裝載於一些常見的嵌入式系統，如自走車、無人機。

二元神經網路的乘積累加運算(MAC, Multiply–ACcumulate)可用三個階段表達[1]。假設有九組權重($w_k, k \in [1,9], k \in N$)和激活值($a_k, k \in [1,9], k \in N$)的輸入要進行乘積累加運算。

第一個階段是乘法，如(1)所示，由於輸入為二元值，其乘法如表一所列，故我們可以使用 XNOR(~^)來對權重和激活值進行乘法運算。

$$o_k = w_k \sim\hat{}\, a_k, k \in [1,9], k \in N \quad (1)$$

w	*a*	*o*
-1	-1	1
-1	1	-1
1	-1	-1
1	1	1

表一、二元值的乘法結果，正是 XNOR 邏輯

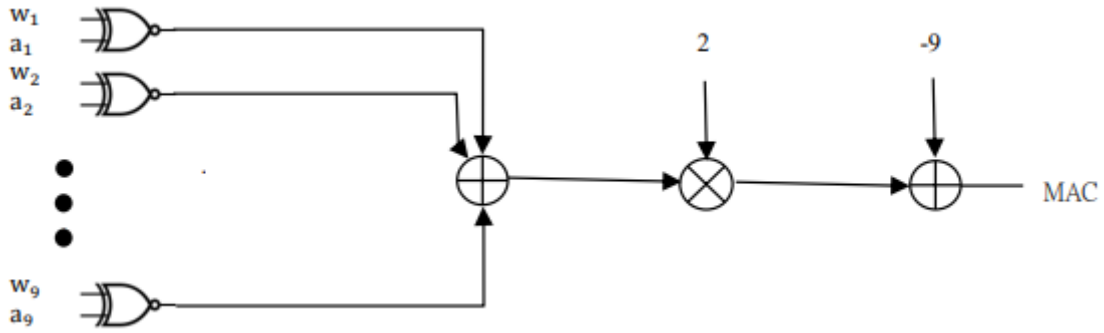
第二個階段是加法之一，如(2)所示，此階段僅算出加法之其中一部分，原因為在電路的角度，0 代表著-1，所以全部加起來-1 反而沒被扣到，因此 s 僅代表九個乘積中，有幾個 1。

$$s = \sum_{k} o_k \tag{2}$$

第三個階段是加法之二，如(3)所示，因為有 9 - s 個-1，故真正的乘積累加運算須修正此項。

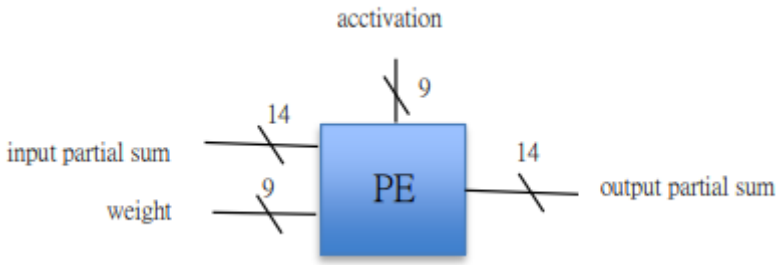
$$\mathrm{MAC} = s - (9 - s) = 2s - 9 \tag{3}$$

二元神經網路之乘積累加運算的電路圖，如圖一所示。



圖一、二元神經網路之乘積累加運算的電路圖

其方塊圖(Block Diagram)如圖二表示，PE 意指處理元件(Processing Element)。其輸入包含寬度(Width)為 9 的權重與激活值以及寬度為 14 的輸入部分和(Input Partial Sum)，其輸出為乘積累加運算結果加上輸入部分和的輸出部分和(Output Partial Sum)。

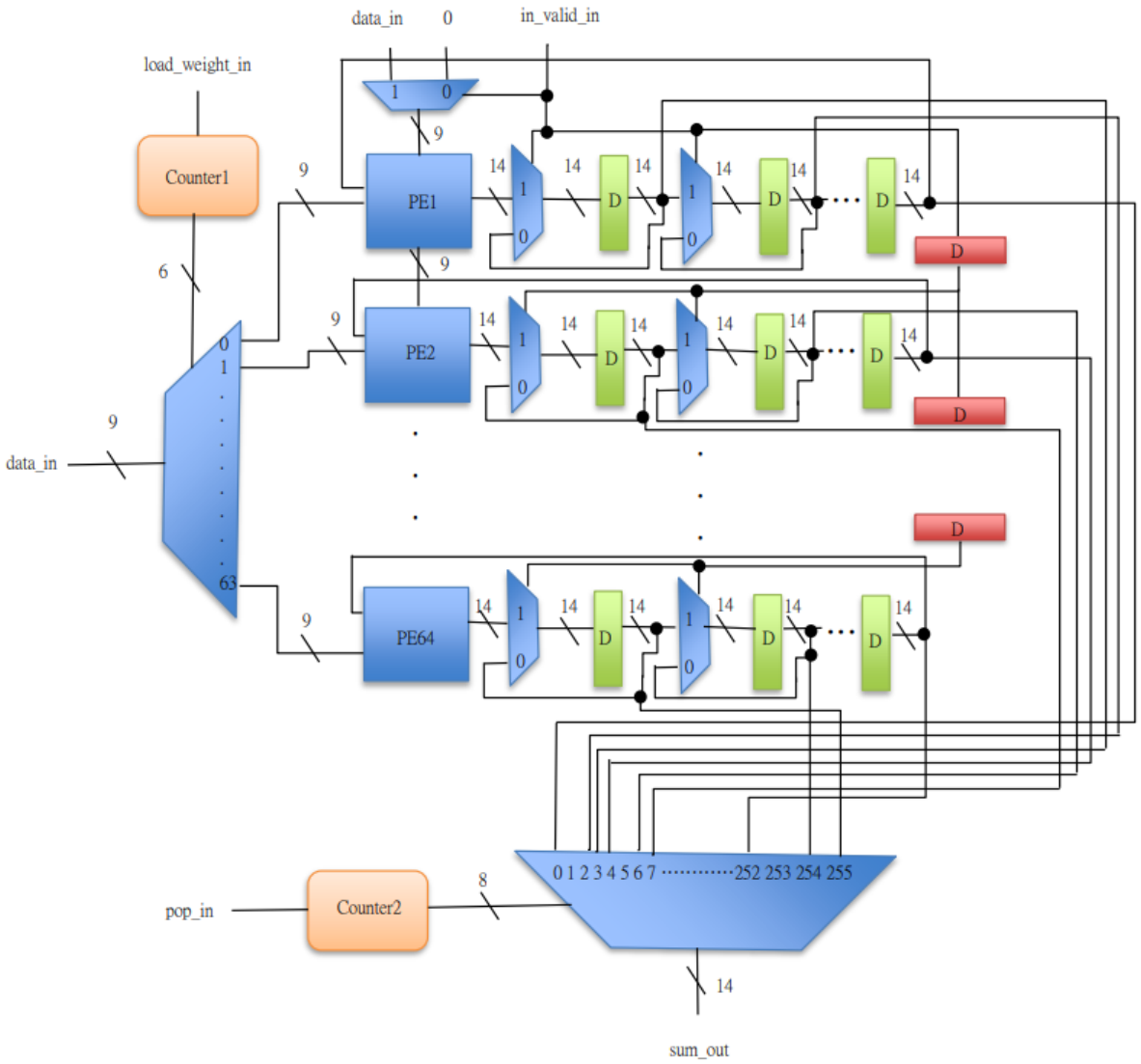


圖二、二元神經網路之處理元件方塊圖

在經過多版的開發後，我們持續優化資料重複使用次數以減少晶片與外界的溝通次數。我們先來介紹我們的**最終設計**。表二為輸入與輸出腳位表、架構圖則放在圖三，此設計由我們小組開發，融合權重固定 (WS, Weight Stationary)與輸出固定(OS, Output Stationary)兩資料流[2]之優點，故名為**權重固定與輸出固定資料流混合設計** (Mixed WS OS Design)。

腳位名稱	寬度	描述
clk_in	1	100MHz 時脈輸入腳位。
rst_in	1	同步低電位有效重設(Synchronous Active Low Reset)輸入腳位。每完成並取出一批計算結果後，藉由重設來讓儲存元件歸零並開始計算下一批計算結果。
data_in	9	權重輸入腳位或激活值輸入腳位。
load_weight_in	1	權重有效訊號輸入腳位。當此訊號為高電位，將 data_in 的輸入存入電路中並代表權重。
in_valid_in	1	激活值有效訊號輸入腳位。當此訊號為高電位，將 data_in 的輸入作為激活值傳入。
pop_in	1	取值訊號輸入腳位。當此訊號為高電位，將計算結果從 sum_out 傳出。
sum_out	14	計算結果輸出腳位。

表二、輸入與輸出腳位表



圖三、設計架構圖

時間線圖如圖四所示，我們先藉由 reset_in 歸零所有圖三中綠色的 D 型正反器(D Flip Flop)，因為綠色的 D 型正反器是我們的部分和儲存元件，從此開始我們便可以拉起 load_weight_in 腳位並開始輸入權重值，當權重輸入完成，拉起 in_valid_in 腳位並開始輸入激活值，激活值會開始由圖三中的 PE1 往下傳遞，此時我們繼續輸入激活值，因為晶片大小之限制，我們決定每一個輸出頻道(圖三中橫列)只放四個儲存元件(綠色的 D 型正反器)，因此我們可以連續輸入四次激活值並放入四個儲存元件，並使用移位暫存器(Shift Register)把 in_valid_in 訊號傳下去，而當一列的 in_valid_in 訊號(紅色的 D 型正反器)是高電位時，才會將計算結果存入儲存元件(綠色的 D 型正反器)，儲存元件也會開始轉動，持續儲存至不同的輸出像素(Output Pixel)。如圖四所示，傳入權重與傳入激活值此兩步驟會持續不間斷之進行以計算新的輸入頻道，得益於我們輸入權重的方式並非使用移位暫存器，當運算還沒結束時我們也能持續更新權重值，也就是當我們結束激活值的輸入，此時，PE1 會完成計算並不會再次用到 PE1 裡面儲存的權

重，故雖然下個時脈週期(Clock Cycle)，PE2 以下還在進行運算，我們已經可以開始更新新的權重值給 PE1 了。當我們完整算完一個輸出像素的值，我們便能藉由 pop_in 的拉起，開始使用 256 個時脈週期把這些數值序列地輸出，與此同時我們可以更新新的權重，因為只要 in_valid_in 沒被拉起來，更新權重並不會影響到儲存元件中之數值。當我們把所有儲存元件之儲存值都取出來之後，我們可以在使用一次 reset_in 訊號來歸零所有儲存元件並開始計算新的輸出像素位置。

圖四、時間線圖(N 的數值由應用有幾個輸入頻道決定)

此外，64 個輸出頻道不一定要全部用到，比如說有 100 個輸出頻道，前 64 個輸出頻道全數計算完後，可以計算剩下 36 個輸出頻道，此時便可以只使用其 36 個橫列，不須輸入完整 64 個權重，而是只輸入 36 個需要的權重即可開始計算，輸出時亦然，只要取完前 36 個橫列的結果即可結束。64 這個數字的決定是因為傳統的神經網路模型通常使用 2 的冪次方作為輸出頻道數量，如此一來可以減少碰到像前述有 64 - 36 個橫列沒被使用到的機會發生。

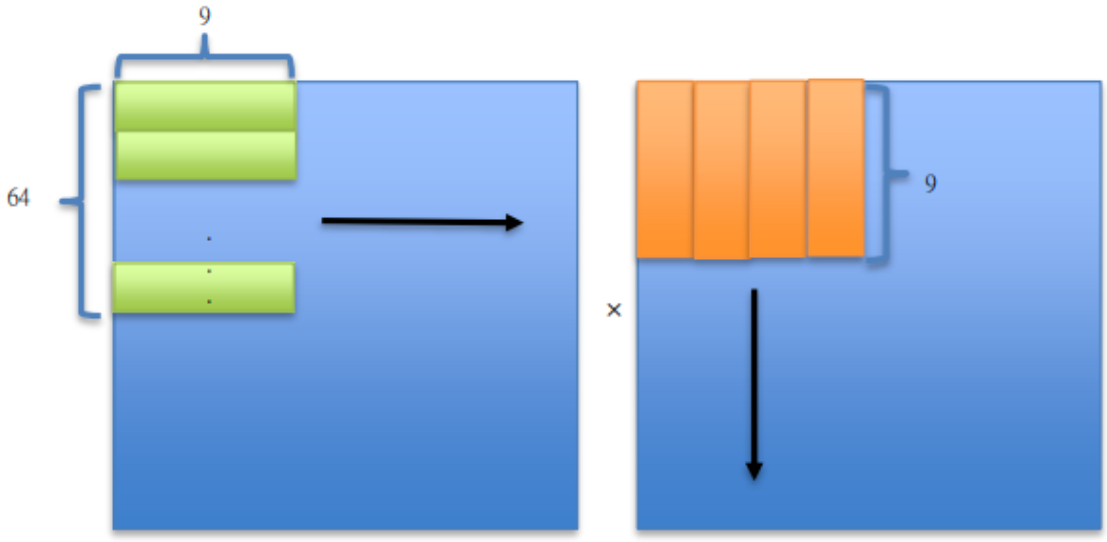
另外，儲存部分和於晶片中這個想法的根源是**輸出固定資料流**，此作法可減少傳統的權重固定脈動陣列(Systolic Array)通常為計算部分和，並輸出部分和，直到不同的輸入頻道權重被載入時，又再次輸入部分和來計算的晶片與外界的通信次數，尤其是當部分和的寬度並不小的時候，這樣的做法反而需要使用更多的時脈週期。

因此，我們的設計是針對二元神經網路之優化。因為二元神經網路的權重與激活值的寬度皆為 1，所以重複輸出與輸入部分和(寬度 = 14)的效率是絕對比較差的。

再來，輸入激活值的地方我們放上一個多工器(Multiplexer)是因為設計考量，考慮在更換權重的 64 個時脈週期，如果這些權重也被當成激活值傳入，那麼雖然因為 in_valid_in 沒被拉起，所以部分和儲存單元不會被錯誤計算值更新到，但這樣的話會增加邏輯閘的切換頻率，使得動態功率增加，故在輸入端放上一個多工器，使得輸入長時間為 0，那麼 PE 們在權重更換前與權重更換後，都會感受到一樣的激活值，故能減少切換頻率。

另外，輸入頻道一次只有一個是有其原因的，也就是為何 PE 只有一行，因為如果我們改成多行的 PE，根據**權重固定資料流**的定義，我們的輸入就得每次都傳給這幾行 PE，也就是假設有 3 行的話，一次就得輸入 27 個激活值，很容易造成剩餘腳位數量受限的問題。故我們最後決定把 PE 做成64×1的大小，而非32×2或其他格式。

此外，此架構仍然支援矩陣運算，並非單純捲積運算加速器，以圖五為例，圖五是兩個矩陣，如果我們要對其執行矩陣乘法的話，我們可以以此方式傳入資料，綠色部分作為權重，橘色部分作為激活值，並在計算完這個區塊後往箭頭方向移動，持續將計算結果儲存至儲存元件內，直到沿箭頭方向計算完成為止。

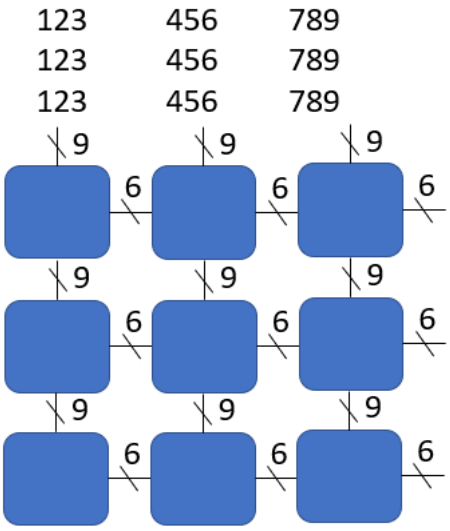


圖五、使用此電路架構於矩陣乘法之示意圖

以下我們將介紹**舊版設計**，並比較最終設計與舊版設計的計算時間、優缺點。

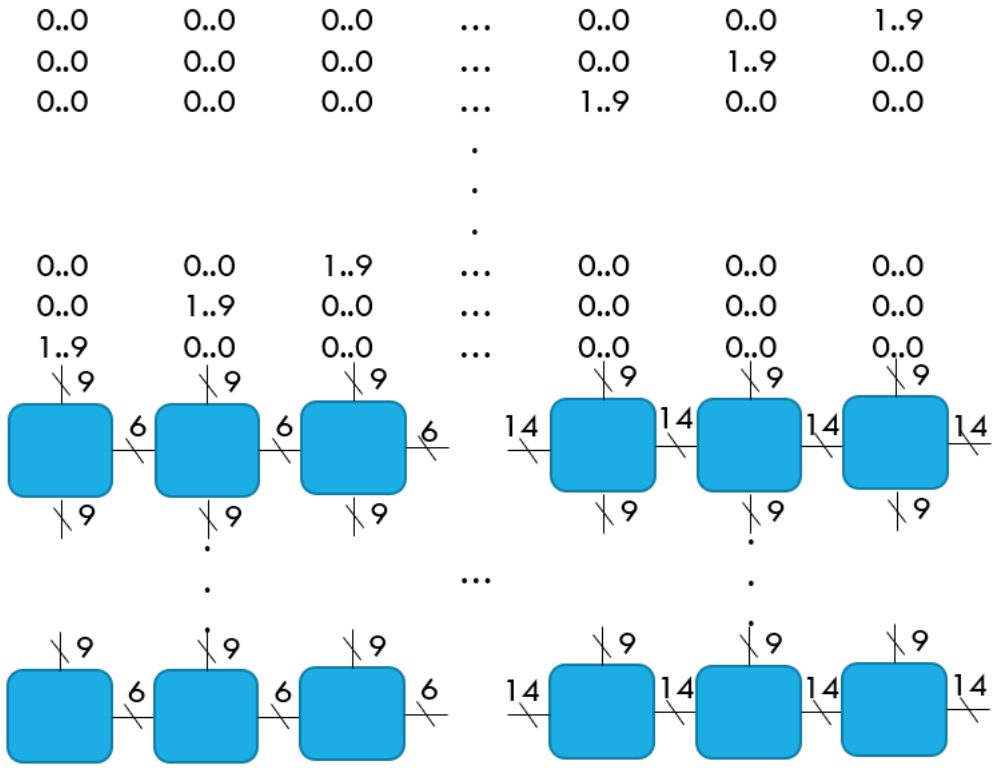
第一版設計

我們第一版設計是考量了我們晶片腳位受限的情況，假設我們只有九個資料輸入腳位，如圖六，我們可以藉由橫列固定(RS, Row Stationary)資料流來做捲積運算。



圖六、以 9 個輸入做捲積運算

而且 9 個輸入也可以作為每個處理元件的權重輸入。此外，我們可以擴大陣列來計算矩陣乘法，如圖七，限於晶片腳位數量，我們無法像權重固定資料流傳遞這麼多的輸入與輸出，故我們輸入一次只會對應到一個處理元件，我們還是藉助權重固定資料流的特性，在輸出頻道的方向重複使用到激活值。

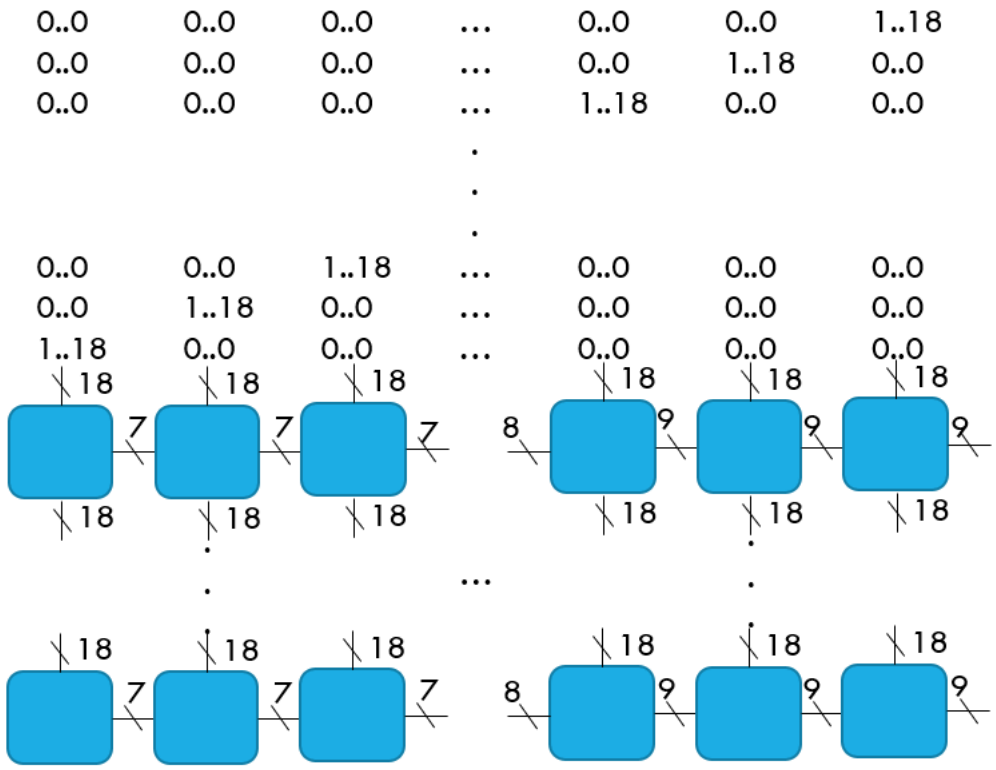


圖七，藉由擴大矩陣來算矩陣乘法

然而，此設計對於捲積運算十分不友善，因為陣列大小只有3 × 3，因此我們決定使用矩陣乘法來計算捲積運算，故第二版設計將只考慮使用矩陣運算的處理。

第二版設計

考量到我們只要做矩陣運算，我們決定加大輸入腳位數量，如圖八，我們使用了 18 個輸入腳位。

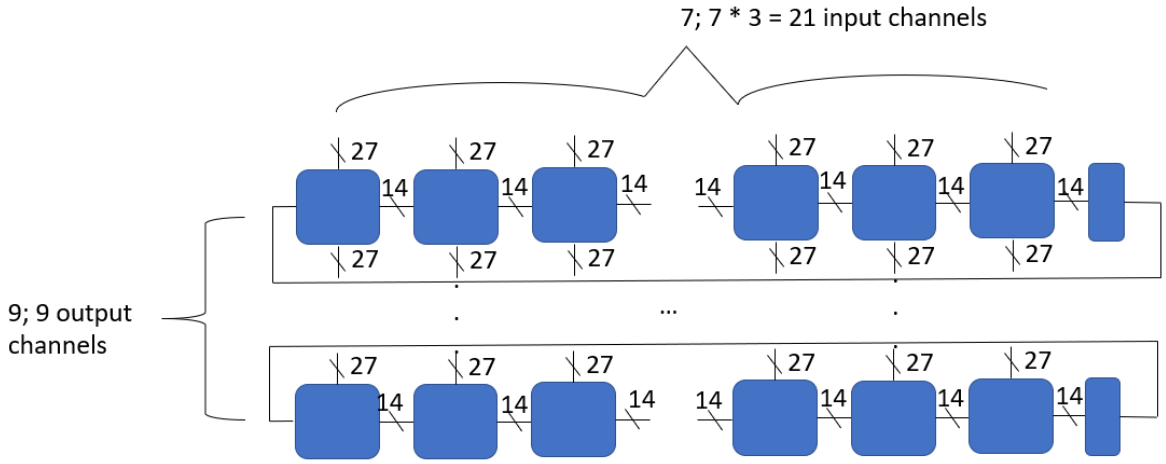


圖八、以 18 個輸入腳位做矩陣運算

然而，我們覺得，既然是二元神經網路，何不輸出一個數值代表是+1 或-1 即好，這樣做的話，晶片外界也不需要處理部分和儲存、累加的事。我們前兩版設計需要外界處理部分和累加，因為我們不希望部分和被來回傳遞。

第三版設計

第三版設計由於我們輸出只需要一個腳位，我們會需要儲存部分和於晶片，直到計算完成再把計算結果輸出。我們將輸入腳位數量開到更大，如圖九，我們有 27 個輸入腳位，希望可以藉由這麼多的輸入腳位來減少傳輸次數，因為 27 個輸入腳位一次就能傳 3 個輸入頻道的激活值。而且，需要取得計算結果的時候，我們可以從儲存元件的正負號直接取得結果。



圖九、藉由更大的輸入腳位來計算矩陣運算

然而，教授提到了一個關鍵問題，為何是 9 個輸出頻道與 21 個輸入頻道?

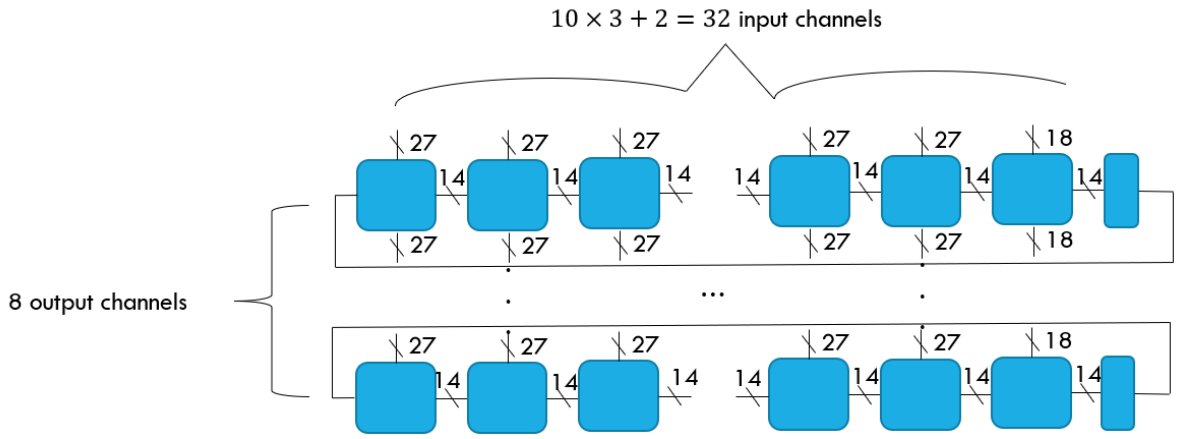
原先的想法是，因為 27 個輸入一次就代表 3 個輸入頻道，要能接近我們常使用的 64 個輸入頻道，因此一次使用 21 個輸入頻道，計算三次就完成 63 個輸入頻道，與 64 接近。但，為什麼不用 24 個輸入頻道，只是多出來的頻道就傳 0 當激活值呢?理由是因為二原神經網路其實沒有 0 這個數字，我們傳入 0 其實是代表-1，如圖一的乘積累加運算電路，此電路不是用以往的乘法電路來計算乘法，而是用 XNOR 來計算，因此我們無法藉由傳入 0 來取消不需要運算的輸入頻道，而是我們會需要傳入一樣多的 0 跟 1 才能平衡掉我們不需要的輸入頻道。這樣的作法可能使用者比較無法接受，使用者需要知道傳幾次 0 和要傳幾次 1 來取消輸入頻道。

然而，誠如教授所言，63 個輸入頻道的確不好，假設使用者原本訓練好一個模型，我還得要他配合，把 64 都改成 63，128 都改成 126 等，這其實也是會大大降低使用者對這顆晶片的購買意願(雖然這是課程產品，沒有要賣)。

而 9 個輸出頻道只是為了配合前述希望使用者使用 63 的倍數個輸入和輸出頻道的情境，因為 9 能整除 63。

第四版設計

主要是修正第三版設計沒有 2 的冪次方個輸入、輸出頻道的問題，如圖十，我們在 10 個 27 位元輸入的處理元件後面放一個 18 位元輸入的處理元件，總共就是 32 個輸入頻道，而晶片在輸入的時候，如果剛好對到 18 位元輸入的部分則只使用 27 個輸入腳位的末 18 位，然而此版的面積過大，我們也不想讓輸入頻道降到 16 個，於是開始重新設計，並朝向更有效率的方向思考。



圖十、將輸入頻道調到 32，輸出頻道調到 8

而且，此想法後來被發現有錯誤，因為晶片以外，我們使用晶片的客戶可能會想使用批標準化(Batch Normalization)，所以我們需要輸出完整計算數值，而且對於權重的重複使用次數可以藉由增大部分和儲存數量來增進，因此我們開發出最終版的設計(Mixed WS OS Design)。如圖三，為何我們只有一行的處理元件，而非一個8 × 8的陣列?因為重複使用次數一樣，但是如果輸入頻道 = 8，我們會需要多傳入 8 次的激活值才能計算下一個輸出像素，這會造成非常多的時脈週期浪費，故決定使用單行處理元件。

此外，如前述，要求使用者儲存、累加晶片輸出的部分和會對晶片外界的儲存單元、計算單元增加負擔，因此我們僅比較不需要傳遞部分和的第四版設計以及最終設計在一些情境下的時脈周期數。

由於最終設計輸出頻道是 64 個，我們在表三比較輸入頻道與輸出頻道皆為 64 個，輸出像素數量分別為32 × 32、16 × 16、4的情境下的時脈周期數。

	第四版設計	最終版設計	比率
32 × 32	1623048	1114432	1.4564
16 × 16	405768	278656	1.4562
4	6348	4417	1.4372

表三、第四版設計與最終版設計在不同使用情境的時脈周期數比較

如表四所列，以前我們太執著於使用到腳位最大數量，而疏於考量資料重複使用效率，且要求使用者必須輸入完權重後立刻輸入激活值，再立即輸入權重的，如此往復，這都歸咎於太大的輸入腳位數量，以致其他應用腳位的使用受限，故在最終版調整回資料輸入腳位只用 9 個，並且靈活使用其餘腳位當傳送權重、激活值或取值的控制訊號，且因為資料重複使用的效率增加，在運算所需的總時脈周期數反而是勝出的。仔細分析後，有發現時脈周期數差距並非等於資料重複使用次數差距，我們將這個差距的來源歸於第四版設計最大化了使用腳位數量，從輸入腳位這個資源得到了一

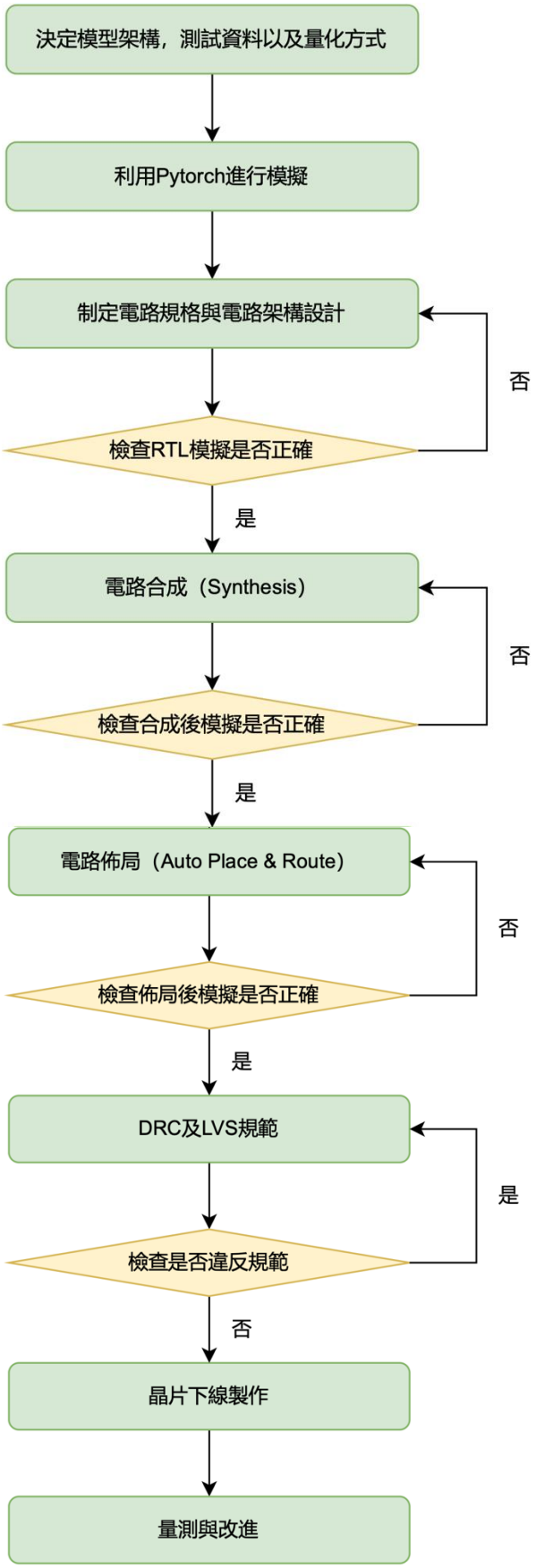
些時脈周期數的減少，比如，輸入三個頻道最終版設計要 3 個時脈週期，然而第四版設計只要一個時脈週期。

	第四版設計	最終版設計
腳位數量多寡	優	劣
腳位使用靈活度	劣	優
資料重複使用次數	劣	優
執行所需時脈週期數	劣	優

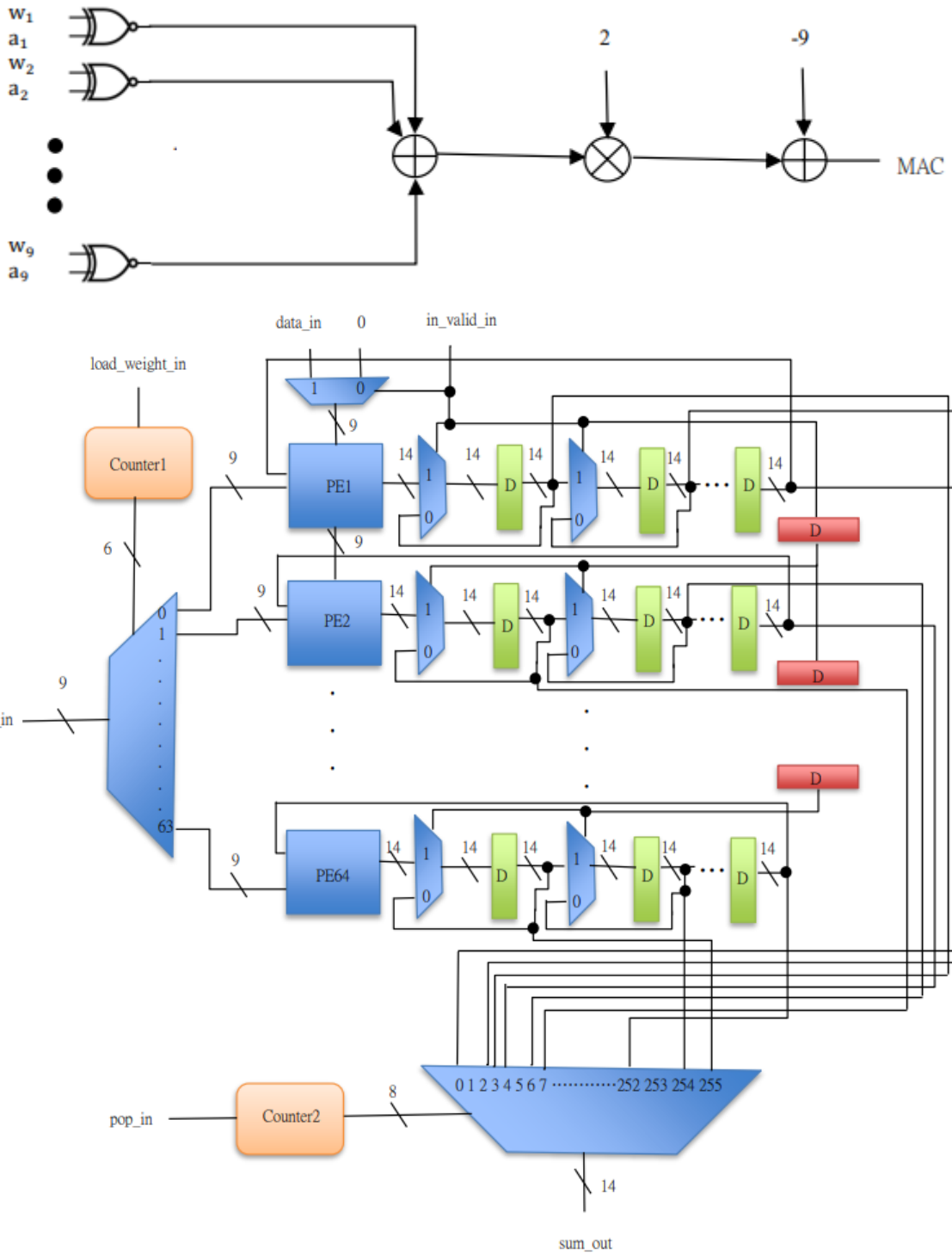
表四、第四版設計與最終版設計的優劣比較

這門課讓我學到了在受限腳位下的設計，在給定腳位數量下，我們不能太貪心於使用所有輸入腳位的資源，反而，我們應該朝向如何更有效率計算的方向思考，適時的去網路上補充一些知識，像是去了解 WS 跟 OS 的優點，並融入我們的整體架構設計，是我們這學期對二元神經網路加速器的解方。

[5] 設計流程


決定模型架構，測試資料以及量化方式
利用Pytorch進行模擬
制定電路規格與電路架構設計
否
檢查RTL模擬是否正確
是
電路合成（Synthesis）
否
檢查合成後模擬是否正確
是
電路佈局（Auto Place & Route）
否
檢查佈局後模擬是否正確
是
DRC及LVS規範
是
檢查是否違反規範
否
晶片下線製作
量測與改進

[6] 電路詳圖

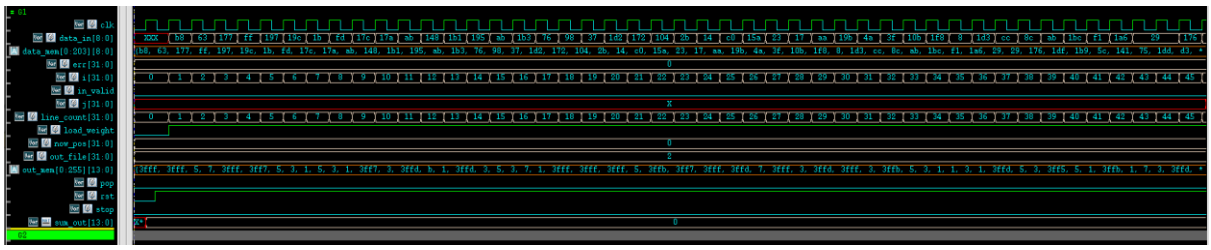

w1
a1
w2
a2
w9
a9
2
-9
MAC
data_in
0
in_valid_in
load_weight_in
Counter1
PE1
PE2
PE64
D
data_in
pop_in
Counter2
0 1 2 3 4 5 6 7 ····· 252 253 254 255
sum_out

[7] 模擬結果

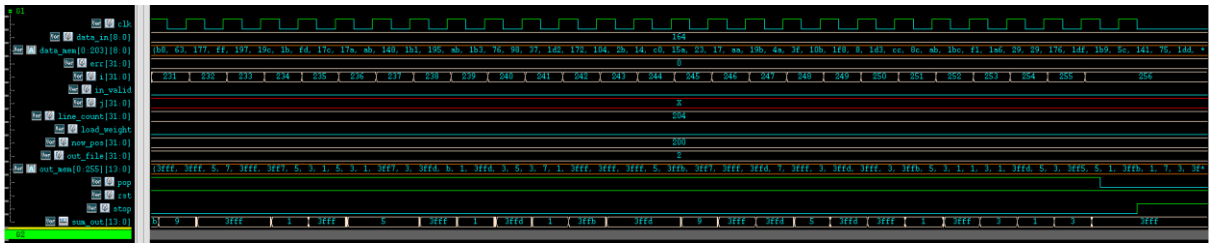
以下是測資(Testbench)在繞線之前、繞線之後的波形圖，取最前面幾個與最後面幾個時脈周期作驗證。

1. Pre-Layout Simulation

最前幾個時脈周期:

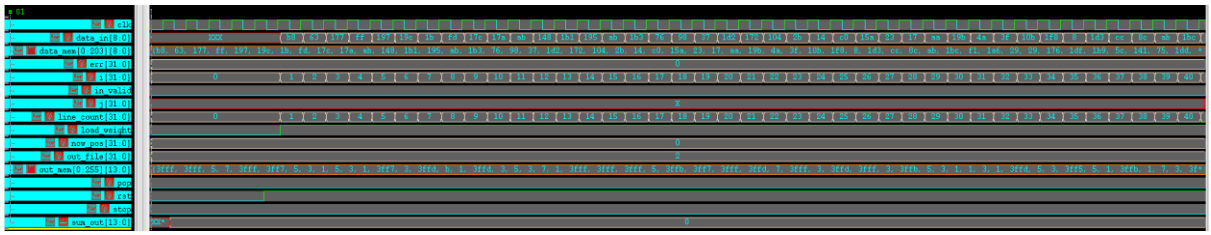


最後幾個時脈周期:

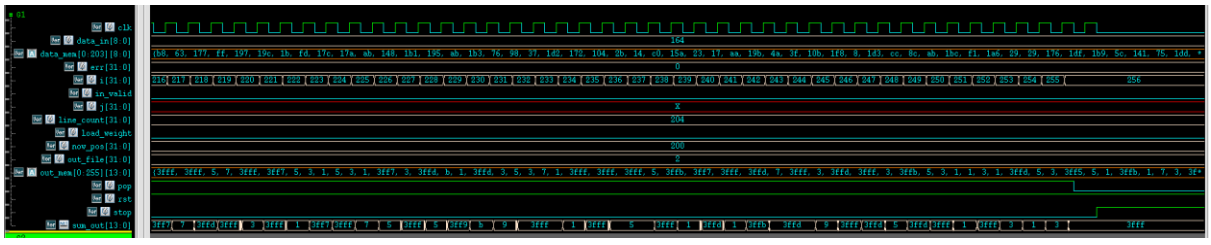


2. Post-Layout Simulation

最前幾個時脈周期:



最後幾個時脈周期:



Post-Layout Simulation 的結果與 Pre-Layout Simulation 相同，故繞線完的晶片是正確的，此外我們也有直接用輸出之計算結果檢查比對過正確答案。

[8] 量測流程

(1) 利用 Pytorch 模擬捲積神經電路，並將其中的權重、激活值、計算結果寫成測試資料檔案。

(2) 電源供應器調整為 1.8V/3.3V 直流電，並接上晶片的 VDD 腳位

(3) 將晶片的 VSS 腳位接地

(4) 訊號產生器調整為 3.3V，產生 100MHz 的方波，輸入 clk 腳位

(5) 將邏輯分析儀接至晶片的輸出腳位

(6) 訊號產生器調整為 3.3V，接至晶片的輸入腳位，向晶片輸入測資並記錄輸出資料

(7) 將晶片輸出與(1)的模擬結果比對，確認是否一致

[9] 佈局驗證結果錯誤說明

1. DRC 驗證結果

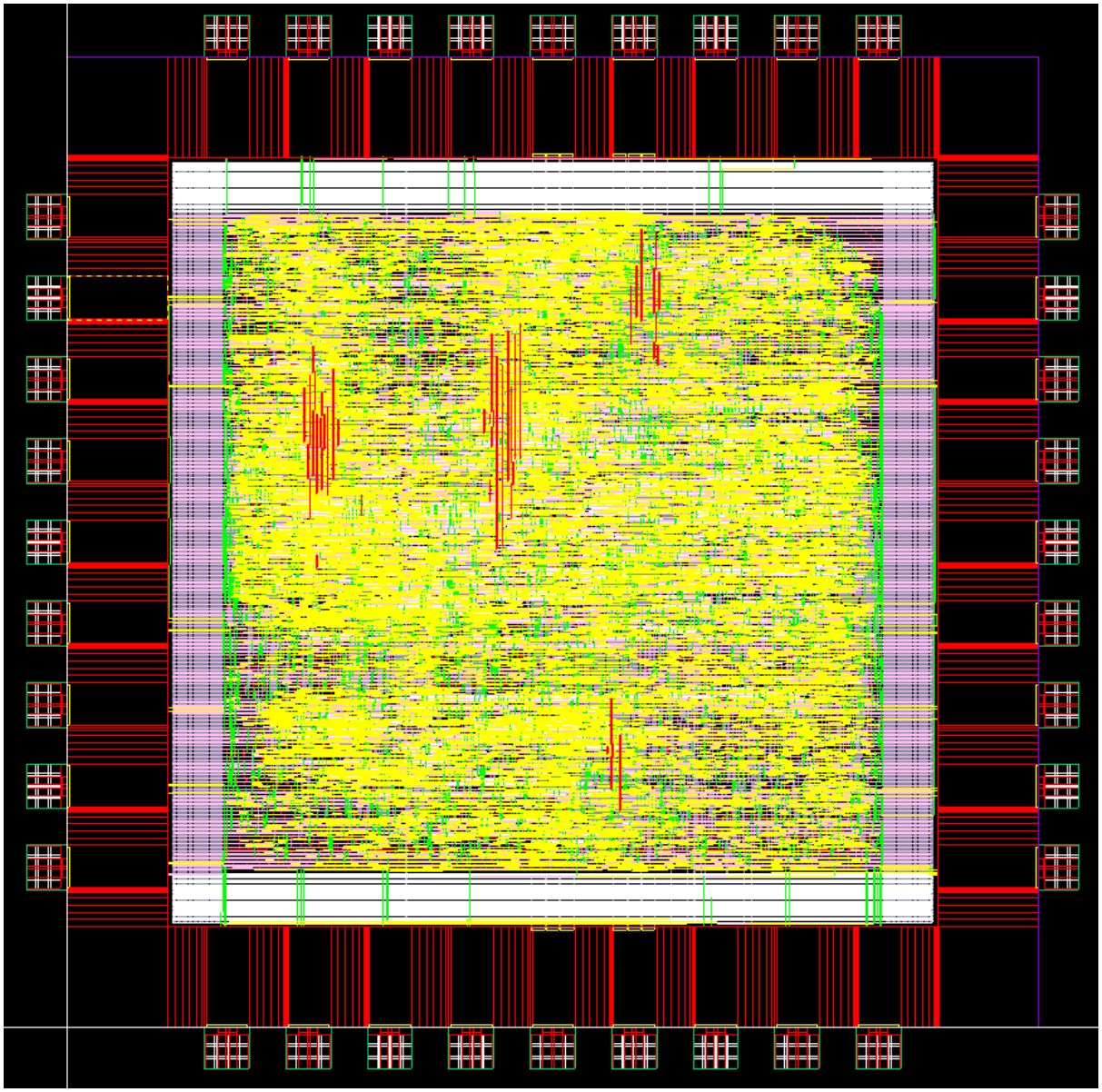
共找到錯誤如下，且皆為允許之 DRC 假錯。

RULECHECK ANT.3.1.1D.ME3 ..... TOTAL Result Count = 6 (6)
RULECHECK ANT.3.1.1D.ME4 ..... TOTAL Result Count = 6 (6)
RULECHECK ANT.3.1.2.Note2.VI5 ..... TOTAL Result Count = 1000 (8768)
RULECHECK RECOMMEND_4.14L ..... TOTAL Result Count = 18 (2529)
RULECHECK 4.29NOTICE ..... TOTAL Result Count = 1 (1)
RULECHECK 4.14Z.NO_IND_PO1 ..... TOTAL Result Count = 1 (1)
RULECHECK 4.22F.NO_IND_M2 ..... TOTAL Result Count = 1 (1)
RULECHECK 4.22G ..... TOTAL Result Count = 1 (1)
RULECHECK 4.24G ..... TOTAL Result Count = 1 (1)
RULECHECK 4.26G ..... TOTAL Result Count = 1 (1)
RULECHECK 4.28G ..... TOTAL Result Count = 1 (1)
RULECHECK 4.31F ..... TOTAL Result Count = 1 (1)
RULECHECK sanity_1 ..... TOTAL Result Count = 16 (448)
RULECHECK IO5.1.W2 ..... TOTAL Result Count = 2 (56)
RULECHECK IO5.1.R1 ..... TOTAL Result Count = 32 (896)
RULECHECK IO5.2.2.L1.a ... TOTAL Result Count = 16 (448)
RULECHECK IO5.2.2.L1.c ... TOTAL Result Count = 16 (448)
RULECHECK Latch.4.1 ..... TOTAL Result Count = 18 (296)
RULECHECK Latch.4.2 ..... TOTAL Result Count = 34 (328)
RULECHECK Latch.4.4.pick ..... TOTAL Result Count = 5 (72)
RULECHECK Latch.4.5.pick ..... TOTAL Result Count = 23 (542)
RULECHECK Latch.4.6.guard ..... TOTAL Result Count = 17 (468)
RULECHECK Latch.4.7 ..... TOTAL Result Count = 53 (586)
RULECHECK Latch.4.7.guard ..... TOTAL Result Count = 5 (72)
RULECHECK Latch.4.8__Latch.4.9__Latch.5.2 ... TOTAL Result Count = 1000 (54762)
RULECHECK Latch.4.10 ..... TOTAL Result Count = 8 (38)
RULECHECK Latch.5.1 ..... TOTAL Result Count = 4 (56)
RULECHECK Latch.5.5 ..... TOTAL Result Count = 124 (856)
RULECHECK Latch.5.6 ..... TOTAL Result Count = 92 (1240)

2. LVS 驗證結果

```
                ################################################
                ##                                            ##
                ##        C A L I B R E   S Y S T E M         ##
                ##                                            ##
                ##            L V S   R E P O R T             ##
                ##                                            ##
                ################################################



REPORT FILE NAME:         lvs_test.rep
LAYOUT NAME:              svdb/CHIP.sp ('CHIP')
SOURCE NAME:              CHIP.spi ('CHIP')
RULE FILE:                G-DF-MIXED_MODE_RFCMOS18-1.8V_3.3V-1P6M-MMC_CALIBRE-LVS-2.1-P8.txt
RULE FILE TITLE:          LVS of UMC 0.18um 1.8V/3.3V 1P6M MMC Mixed Mode/RFCMOS Process
HCELL FILE:               (-automatch)
CREATION TIME:            Wed May 24 10:55:46 2023
CURRENT DIRECTORY:        /home/raid7_2/userb07/b7502027/APR_final/Lab5_LVS
USER NAME:                b7502027
CALIBRE VERSION:          v2022.3_33.19    Tue Sep 6 12:10:05 PDT 2022


                          OVERALL COMPARISON RESULTS


                     #       ###################          _   _
                    #        #                 #         *   *
                  #   #      #     CORRECT     #           |
                   # #       #                 #         \___/
                    #        ###################
```

[10] 佈局平面圖

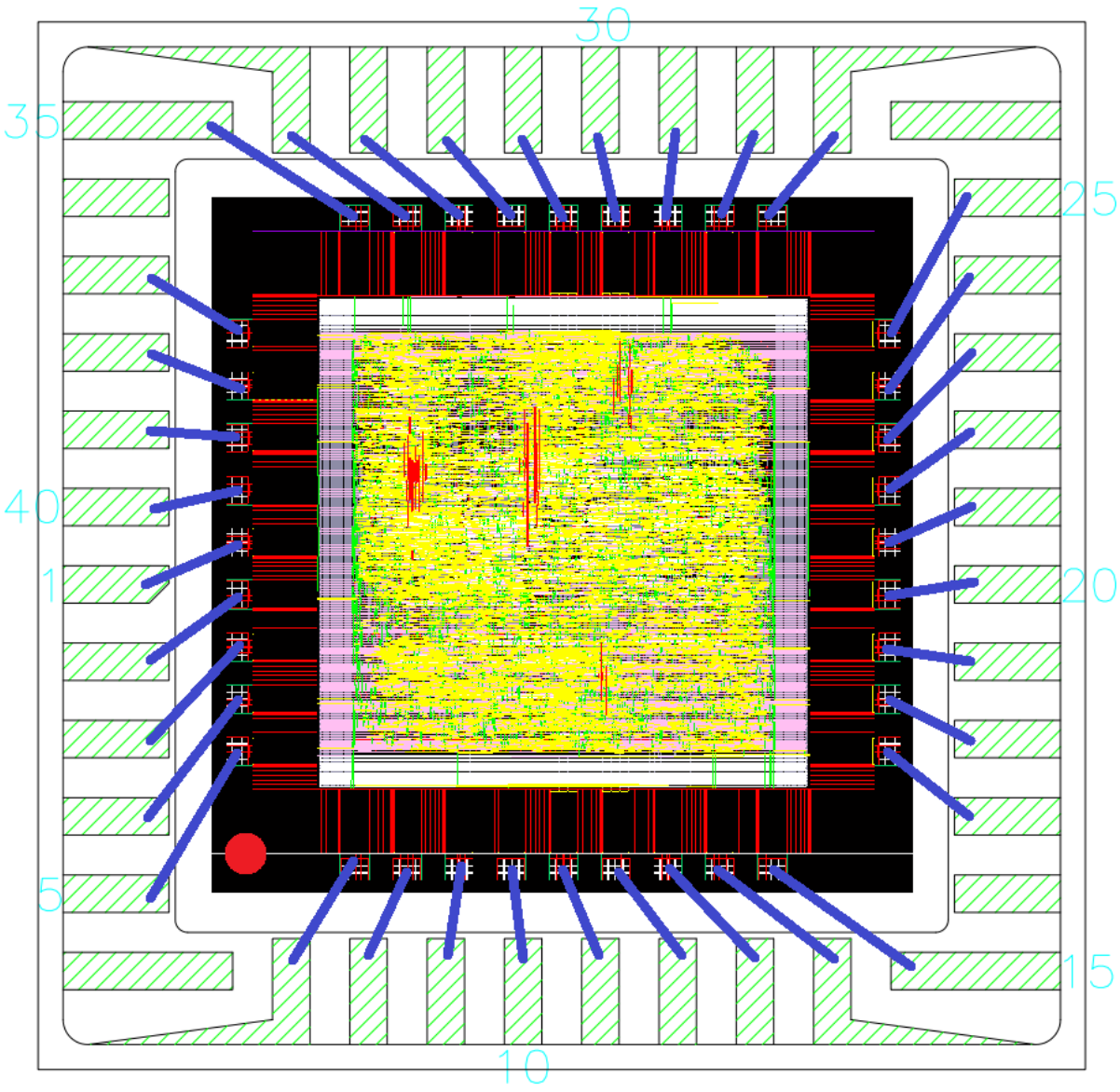


Chip Size：1.46658 mm*1.467 mm

Power Dissipation：84.9830 mW

Max Frequency：100MHz

[11] 打線圖



[12] 預計規格列表

Description	
Process	**UMC 0.18um Mixed-Mode and RFCMOS 1.8V/3.3V 1P6M Metal Metal Capacitor Process**
Power Supply	**1.8V**

Specification	Spec.	Pre-sim	Post-sim
Frequency	**100 MHz**	**100 MHz**	**100 MHz**
Chip size	**< 1.5 mm*1.5 mm**	**515020 μm^2**	**1.46658 mm*1.467 mm**
Power	**-**	**28.3318 mW**	**84.9830 mW**
PADs	**36**	**36**	**36**

[13] 參考文獻

[1] https://ieeexplore.ieee.org/document/10040675/
[2] https://www.researchgate.net/figure/Illustrations-of-a-weight-stationary-and-b-output-stationary-data-flows_fig4_357893531