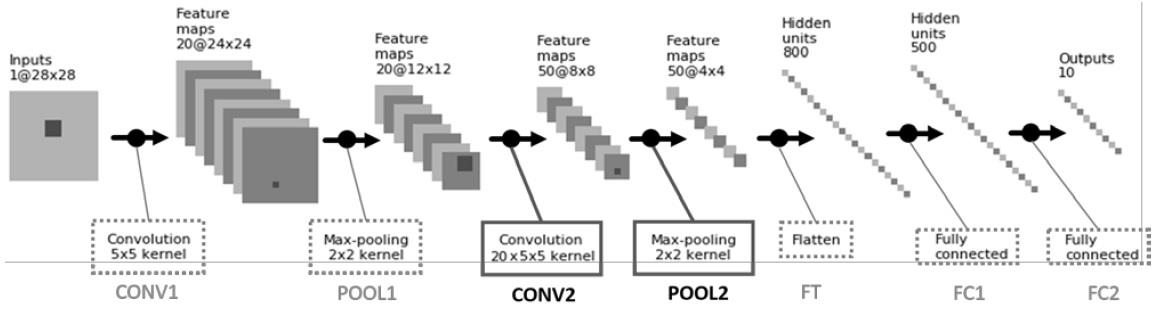
ICLAB Final Report

Redesign Lenet Accelerator

Member : 翁啟文、賴柏光、羅允辰

### **Lenet & MNIST Brief Introduction**

Lenet 是一個擁有兩層 Convolution-Pooling Layer 與兩層 Fully Connected Layer 的架構(如 Fig.1 )，其中，我們 Final Project 使用的 Input data 為 MNIST，是一個由手寫數字 (0~9) 組成的 dataset，將任意一張 MNIST dataset 中的照片與我們預先設定的 Lenet 權重值做運算後，將能以 99% 的正確率辨識出數字為何。



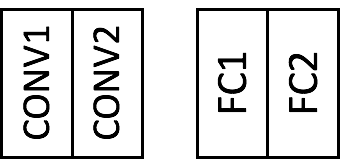
**Fig1. Lenet Architecture**

各層(i.e. Convolution Layer, Max-Pooling Layer, Fully-connected Layer) 的運算細節已在 Interim Report 中詳述。若助教有任何疑問，歡迎參考 Group 7 Interim Report。

### **Hardware Design Consideration**

**Architecture Selection**

硬體架構可以粗略分成以下三種，我們將討論各種設計之優劣，和最後我們的選擇：

1. 每層 Layer 一個硬體 ( e.x. HW5 )

優點：容易設計

缺點：a. 使用太多乘法器使面積很大

b. 每一層 cycle count 相差很大

c. Pipeline 若 cycle count 要近似，

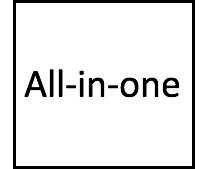
每一層需要的SRAM數目將相差

很大。

d. 若要 pipeline 則會需要四組 SRAM 來

**Fig2. (1) block diagram**

暫存

1. 全部 Layers 在同一個硬體

優點：面積小

缺點：a. Controller 較難設計

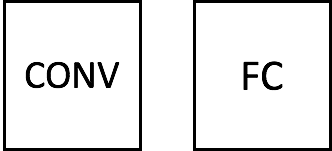
b. 無法 pipeline 運算

**Fig3. (2) block diagram**

1. Convolutions Layers 與 Fully-Connected Layers 各自設計硬體

優點：a. 容易 pipeline

b. 面積與 all-in-one 相差不遠

 c. Controller 設計相對all-in-one容易

缺點：需要一組 SRAM 來暫存 Feature map

**Fig4. (3) block diagram**

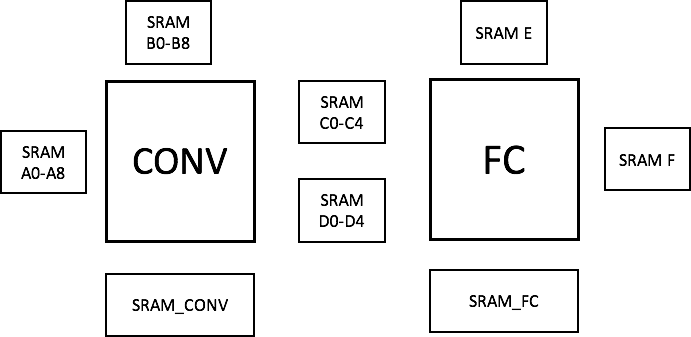
根據以上的分析，我們將 Convolution Layer 與 Fully-Connected Layer 各自設計。原因為：

1. 因為其容易被 pipeline ，能夠有效縮短單張照片運算的 cycle count (當 pipeline 被塞滿後)，在加速器的設計當中，計算出一張照片的 cycle count 是十分重要的考量因素。

2. 面積不會太大。

3. 僅需要多一組 SRAM 來暫存。

### **Our Design**

下圖為我們最終設計的 Block Diagram，包含 8 組 SRAM(A-F + CONV + FC) 與兩個處理單元 (CONV + FC) 。

**Fig5. Our Design Block Diagram**

### **Pipeline Analysis**

我們計算處理 CONV 所花的 cycle 數，在 CONV 中我們使用了 25\*4 = 100 個 8-bits \* 4- bits 的乘法器，因此我們計算出來處理一張圖所花的 cycle 數為：

CONV1 的 cycle 數:約 12\*12\*20 = 2880

CONV2 的 cycle 數:約 4\*4\*20\*50 = 16000

CONV 的總 cycle 數:約 2880 + 16000 = 18880

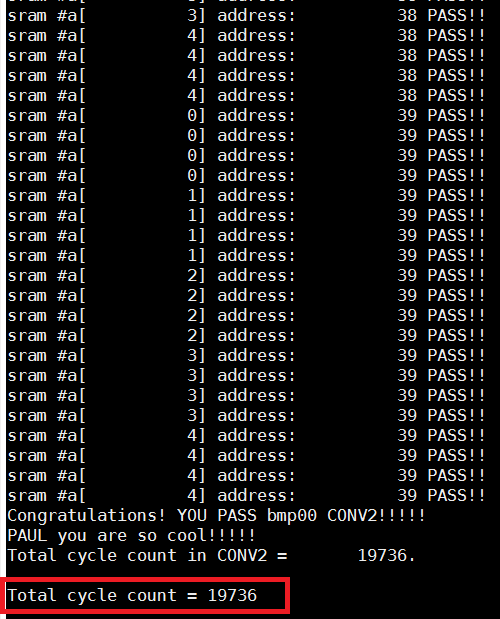
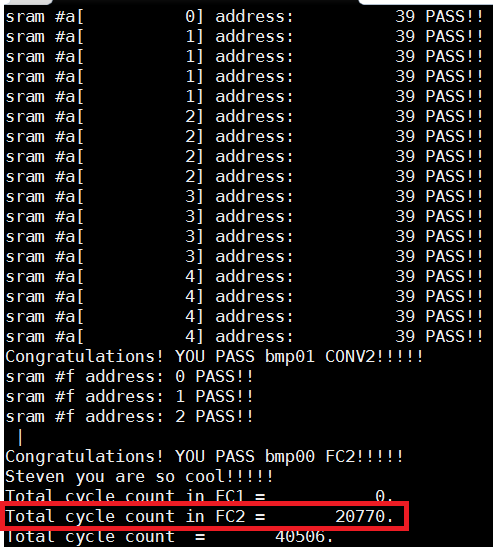
因此我們在設計 FC 時，就盡量讓 FC 處理 fully connected layer 的 cycle 數接近 18880 這個數 字，把 pipeline 切齊。經過討論後，我們決定讓 FC 使用 20 個 8-bits \* 4-bits 的乘法器，而計算 出來處理一張圖 FC 所花的 cycle 數為：

FC1 的 cycle 數:約 800\*500 / 20 = 20000

FC2 的 cycle 數:約 500\*10 / 20 = 250

FC 的總 cycle 數:約 20000+ 250= 20250

經過上面計算後，我們可以知道切成 pipeline 後，圖片出來所花的 cycle 數約為 20250 左右。實際上執行 RTL 的 simulation 也是差不多是這個數字:

**CONV:cycle** 數為 **19736 FC:cycle** 數為 **20770**

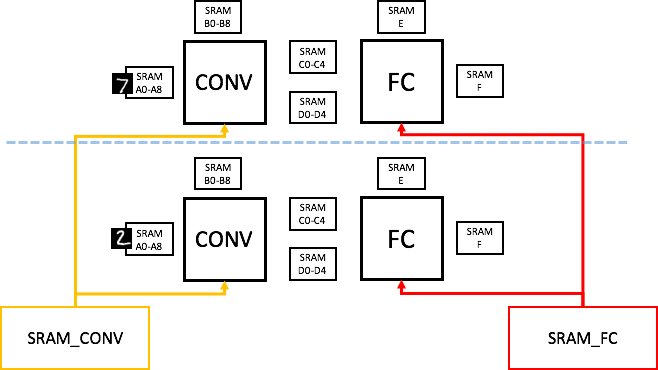
如果只跑一張圖片，所需要的 cycle counts 為 40506，看不出效益;然而，如果跑十幾張、 數百張圖片，則 pipeline 的效益就會顯現出來:平均每張圖片的 cycle counts 就會接近一半，也 就是為 20770。

**我們的設計在與 HW5 比較後，得到 2.108x 的 Speedup**

**面積為 84022.707，為 HW5 的 1/2 倍。**

### **Optimization**

最後，我們為了增加 weight 的使用率，也就是每一個 weight 可以被多少張照片使用，我們設計了 multiple set 硬體共用 weight 這一個機制。如下圖所示，我們將同一份 weight 同時送給兩個硬體，但是兩個硬體的輸入的照片卻是不同張( e.x. 7 and 2 )，我們可以在增加 weights 共用率的同時，增加 Speed Up。



**Fig6. Multiple Sets Sharing Weights**

**我們的最終設計在與 HW5 比較後，得到 4.217x 的 Speedup**

**面積為 149060，僅為 HW5 的 1.13 倍。**

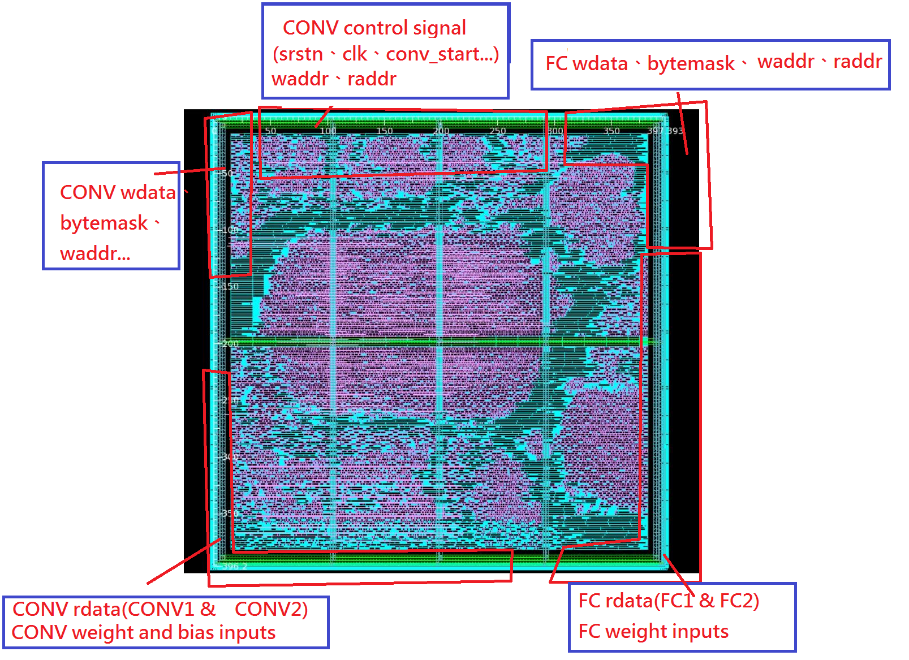
### **Gate-level synthesis and simulation**

合成方面我們原本使用3.5 ns可以滿足set-up time & hold-time requirements，但是在之後P&R無法合成出正確的電路，因此我們將clock period放寬至4 ns，P&R才成功。我們使用Design Compiler合出1-set的LeNet accelerator和2-set的LeNet accelerator，2-set LeNet accelerator area約為1-set的兩倍，礙於篇幅限制，詳細area、cycle time、power等數據可以參考SYN資料夾裡的相關報告。

### **Layout Detail**

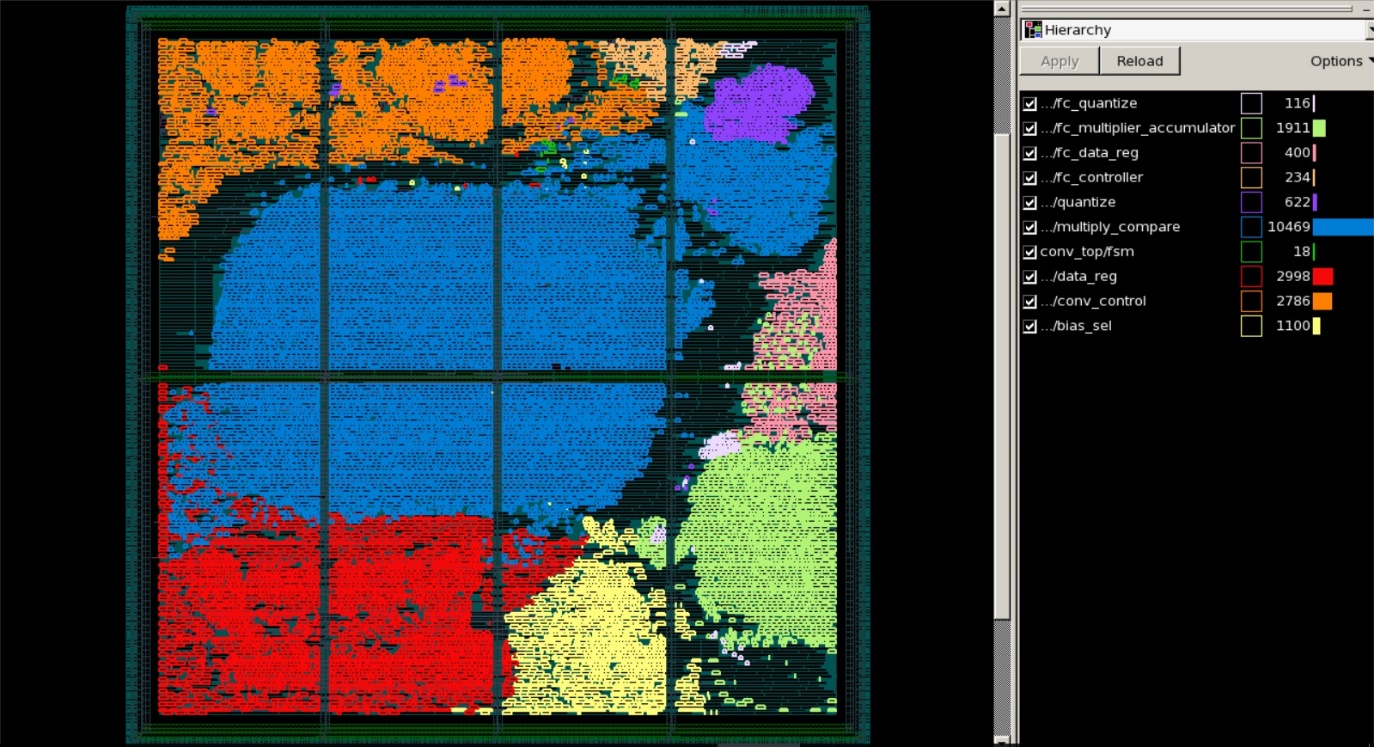
1. I/O pin的配置：

為了讓相關的邏輯放在一起，我們將相關邏輯的I/O pin盡量擺在一起，並平衡四邊的pin位數量。總共有1662條pin位，長有416個pin，寬有415個pin。Chip長約為397.393um，寬約為396.200um，core area為397.393 \* 396.200 = 157447.1066 um2，cell area: 88755.872684 um2，Core utilization約為56%左右。



1. Hierarchy layout view：

各個module在chip layout分布如下：



可以發現，主要是multiply unit佔最大的面積，大約占總體面積的57%左右，因此未來如果要在更進一步優化面積，可以從乘法器單元著手研究。

### **DRC & LVS Verification**

在驗證DRC與LVS後發現，DRC可順利通過，然而，檢查LVS的過程中，出現了VSS與一條wire發生短路的情況。推測可能是在繞線的過程中，PG connection沒能正確設定，導致VSS發生問題，相信在修正指令後，可順利解決這個錯誤。

### **Post-layout Simulation**

根據P&R的結果進行post-layout simulation，經過嘗試錯誤的結果，將clock period設定為4.2 ns (約是合成cycle period 4 ns的**1.05**倍)，儘管post-layout simulation能順利完成，但是會在P&R的過程中產生setup time violation，因此我們放寬test bench中的clock period來避免在post-layout simulation時出現setup time violation，使post-layout simulation不會產生任何問題。