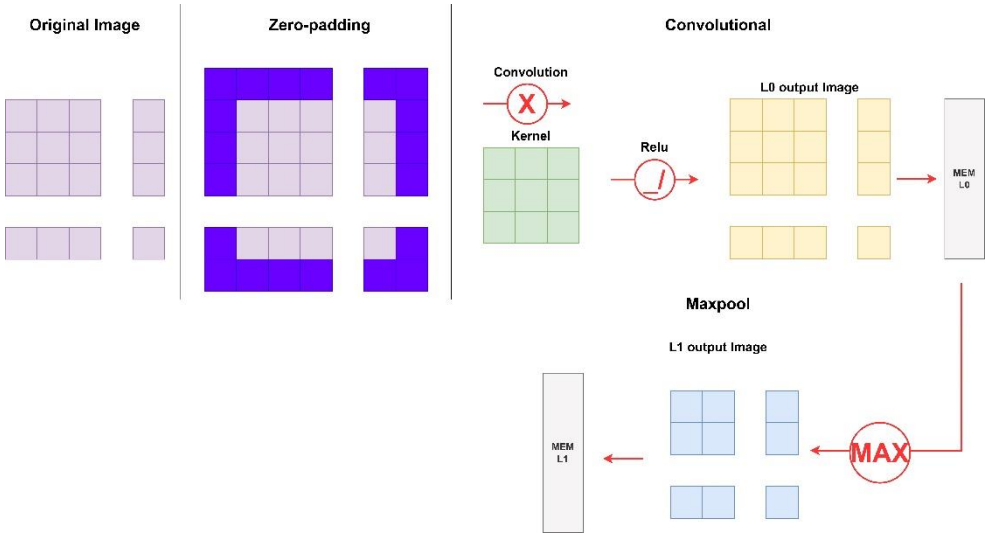
壹、 演算法介紹與說明

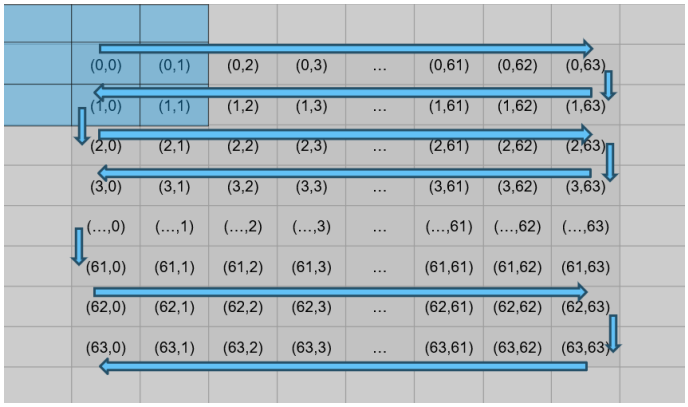
本次期末專題主題為圖像卷積運算特徵提取，基本的流程如題目附圖



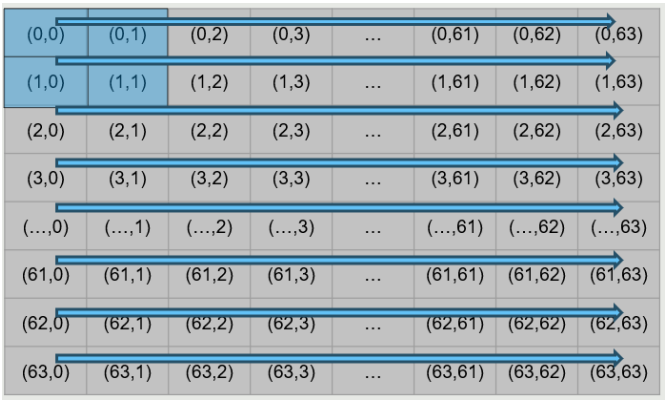
首先，將 64x64 灰階圖片進行 zero padding，圖像外圍補上一圈 0，接著使用 3x3 的 filter 以 stride=1 掃過整張圖像。filter 內的 9 個權重與對應的圖片像素相乘並加總，經 ReLU 激活函數後輸入到記憶體 L0。掃描完成後得到 64x64 的特徵圖。接著，使用 2x2 的 max-pool filter 以 stride=2 掃描特徵圖，輸出 32x32 的縮小特徵圖輸入到 L1 記憶體。為使用硬體實現此捲積運算，需重新設計讀值與 filter 移動等演算法，後續將講解電路設計的兩個版本及其差異。

一、 版本一電路

1. 演算法



▲圖一



▲圖二

（1）Zero-padding 演算法

(圖一)顯示 64x64 灰階圖像經過 zero padding 的示意圖，藍色框為 filter 位置，箭頭代表移動方向。第一版演算法中，九個暫存器記錄 filter 當前九個位置的灰階值，count_y 和 count_x 計數器記錄 filter 中心位置，共需 6x4096 個 cycle 完成運算。filter 移動方式利用位移暫存判斷方向，三個 cycle 更新資料，zero padding 則檢測是否在邊界並輸入 0。

（2）乘加及 ReLU 演算法

移動 filter 時，三個 cycle 輸入 idata 值，採用 pipeline 方式加速。每列數值相乘相加後更新結果，依第 40 位判斷輸出灰階值或 0，並取中間 20 位數與第 16 位相加以完成四捨五入計算。

（3）Max-pooling 演算法

(圖二)為 64x64 原始灰階圖像經卷積後的特徵圖，藍色部分為 max-pool filter，箭頭表示移動方向。使用四個暫存器取出 L0 中特徵圖數據需 4 個 cycle，並用三個比較器選出最大灰階值輸入 L1，共需 6x1024 個 cycle 完成 max-pooling 運算。因 stride=2，每次更新需更換暫存器所有值，當 count_x=62 時歸零，count_y 加 2，定位於 filter 左上角。

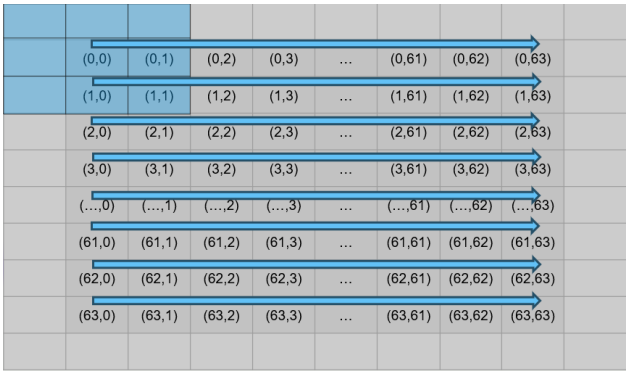
2. 演算法優缺點:

第一種演算法優勢在於整體需要的 cycle 數少理想情況僅需 30000 出頭並且使用到的九個暫存器在做兩部分運算時皆能夠很好被重複利用。缺點則是為了完成此演算法，需要使用到三個 20bit 乘法器與最少三個 40bit 加

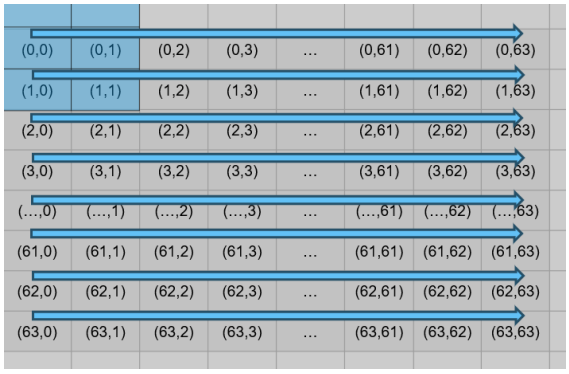
法器。在經過實作合成電路後，我們發現第一種演算法 cycle 數優勢無法打平第二種演算法的面積優勢。

二、 版本二電路

1. 演算法:



▲圖三



▲圖四

（1）Zero-padding 演算法

(圖三)為 64x64 灰階圖像經 zero padding 示意圖，藍色框為 filter 位置，箭頭表示移動方向。第二版演算法中，仍使用 count_y 和 count_x 計數器記錄 filter 中心位置，新增 count_pixel 花九個 cycle 依序輸入九筆灰階值。filter 根據座標(6’dy, 6’dx)移動，zero padding 檢測邊界並控制暫存器輸入 0。

（2）乘加及 ReLU 演算法

運算需九個 cycle 輸入九筆灰階值，採用 pipeline 算法，資料輸入後執行乘法並傳至累加電路，累加 9 次後根據第 40 位判斷輸出灰階值或 0，取中間 20 位數與第 16 位相加完成四捨五入。此算法共需約 13x4096 個 cycle 完成 convolution，其中前 9 個 cycle 取 idata，後續依次進行相乘、相加、ReLU、寫入 L0。

（3）Max-pooling 演算法

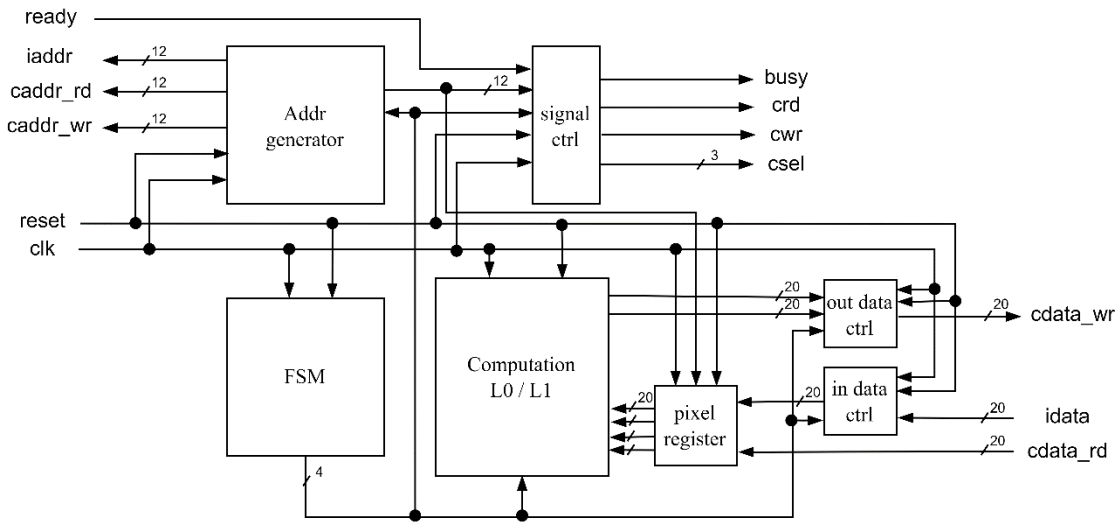
(圖四)為 64×64 灰階圖像經卷積後的特徵圖，藍色部分為 max-pool filter，箭頭表示移動方向。使用一個暫存器取出 L0 數據需 4 個 cycle，並用一個比較器比較當前輸入與先前最大值，最終輸出最大灰階值至 L1。因 stride=2，每次更新需更換暫存器所有值，當 count_x_max=62 時歸零，count_y_max 加 2，定位於 filter 左上角。

2. 演算法優缺點:

此種演算法的優勢在以電路實現僅需一個乘法器及加法器，並且使用 pipeline 壓縮運算時間，使得總運算時間不會超過第一種演算法太多。最終選擇此演算法為我們的最終版本。

零、 電路架構

一、 版本一電路



▲圖一

圖一為我們最初設計之電路架構，設計演算法以上已詳述。模組方面，我們使用到了 FSM 進行狀態轉移，並使用 pixel register 進行像素的暫存。以下我們詳細介紹。

（一）FSM：

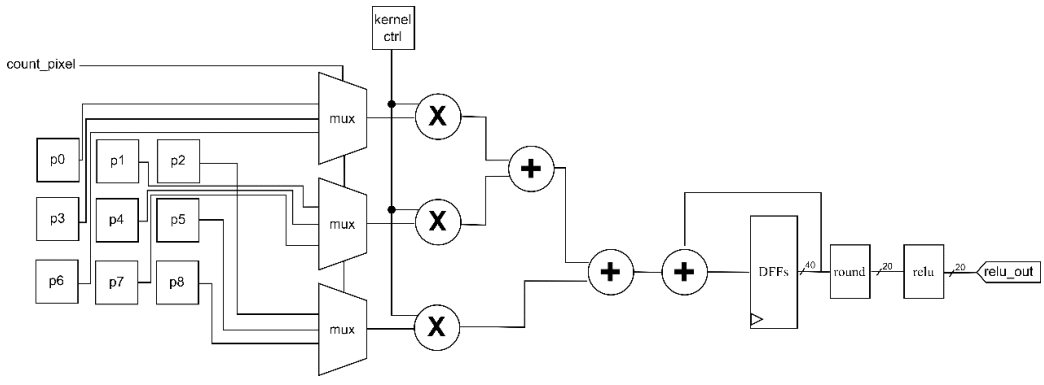
本架構使用到 9 個狀態轉移，產生訊號控制整個電路運作處理。

（二）Addr generator：

此版本電路計算位址只需要一個計數器 counter，透過其定位產生讀取資料的位址和寫入資料之位址。設計方法運算之安排，我們是先算完 L0 再算 L1，因此計數器會重複運算，優點是能共用硬體架構。

（三）Computation L0 / L1：

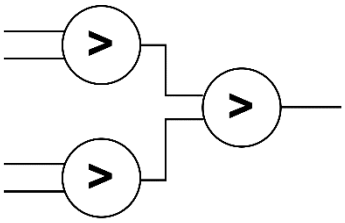
1. L0 計算：



▲圖一 a

此模組負責計算 L0 之卷積數值，方法如演算法所述，我們需要 9 個暫存器，透過一位暫存 方法存入，並透過多工器選擇每次運算的 pixel 為何，多工器控制訊號為 count_pixel，分三次將所有數值累加運算完，再進行 round 和 relu。

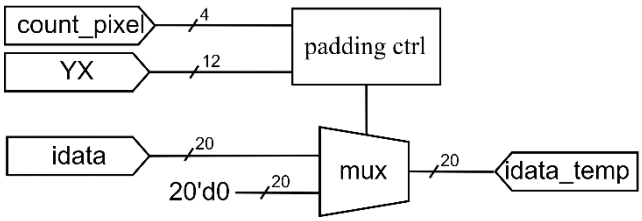
2. L1 計算：



▲圖一 b

此模組負責計算 L1 之 maxpool 數值，為 combinational 電路，一次輸入四筆數值，比較後輸出。

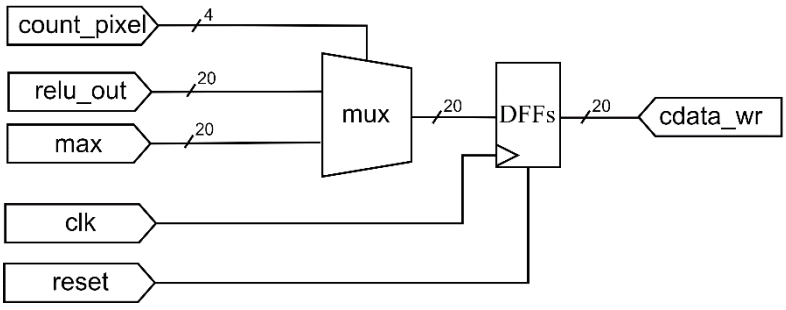
（四）in data ctrl（input data control）：



▲圖一 c

這個模塊會根據位址 counter 定位 YX，決定是否執行 zero padding。要的話，輸入 0，否則輸入 idata。

（五）out data ctrl（output data control）：



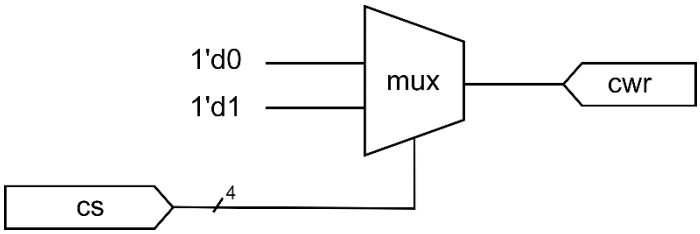
▲圖一 d

此模塊控制輸出 cdata_wr 為 L0（relu_out）還是 L1（max）的運算數值，我們一樣使用 count_pixel 控制多工器，先經過暫存再輸出。

（六）pixel register：

此模塊即為 9 個 DFFs，在 L0 時使用全部暫存，在 L1 時使用 4 個暫存，達到硬體共用之目的。

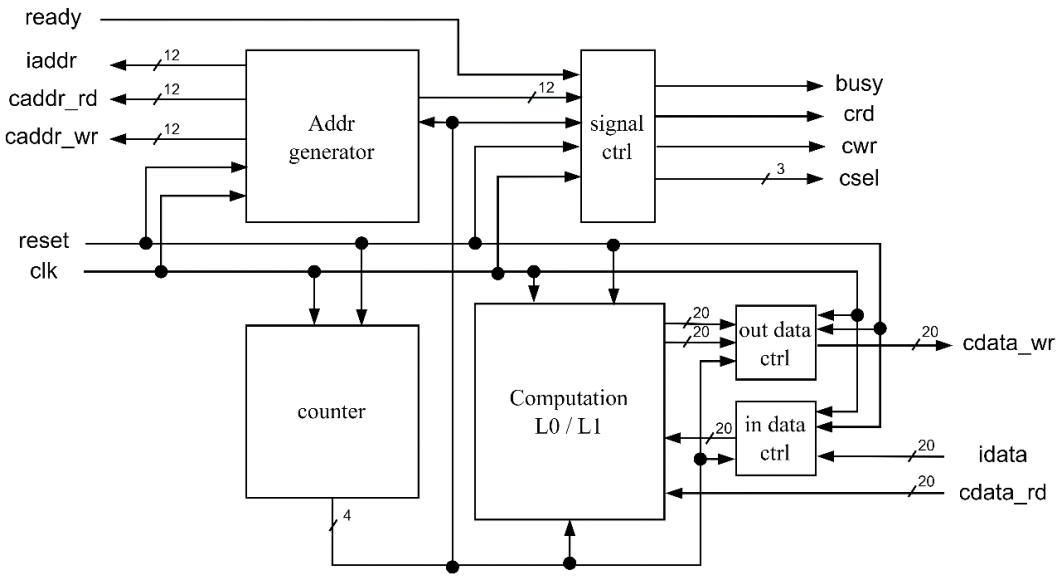
（七）signal ctrl（signal control）：



▲圖一 e

對應圖一中的 signal ctrl，此模塊產生 busy、cwr、crd、csel 之控制訊號，此模塊為 combinational 電路，內部架構皆相似，以 mux 決定輸出數值，控制訊號為 count_pixel。以 cwr 為例，當 cs（current state）或 count_pixel 為對應數值則輸出 0 或 1。

二、 版本二電路：

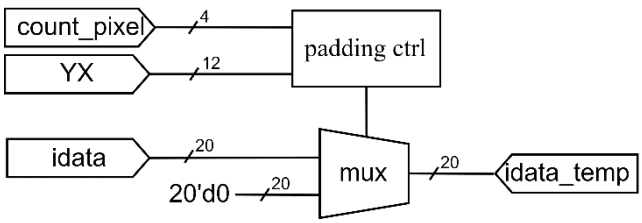


▲圖二

（一）counter：

首先，我們使用一個 counter 計數，從 0 到 12，前方以詳述過，此 counter 產生多數模塊的控制訊號。

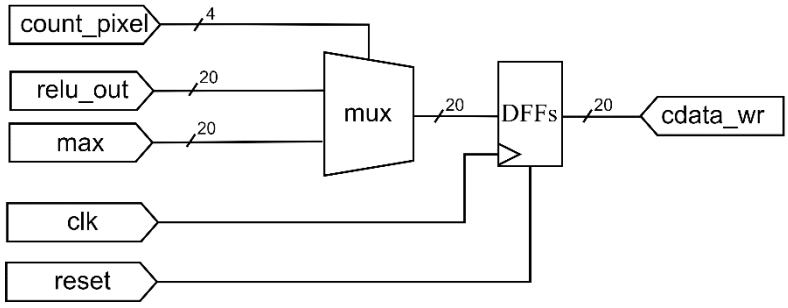
（二）in data ctrl（input data control）：



▲圖二 a

此模塊控制輸入卷積的數值為何，而 Padding ctrl 是一個藉由 count pixel 控制的 combinational 電路，他在不同的 count pixel 時判斷 counter XY 的定位，再決定要不要執行 zero padding，其產生一控制訊號控制多工器輸入捲積計算為零或 idata。

（三）out data ctrl（output data control）：

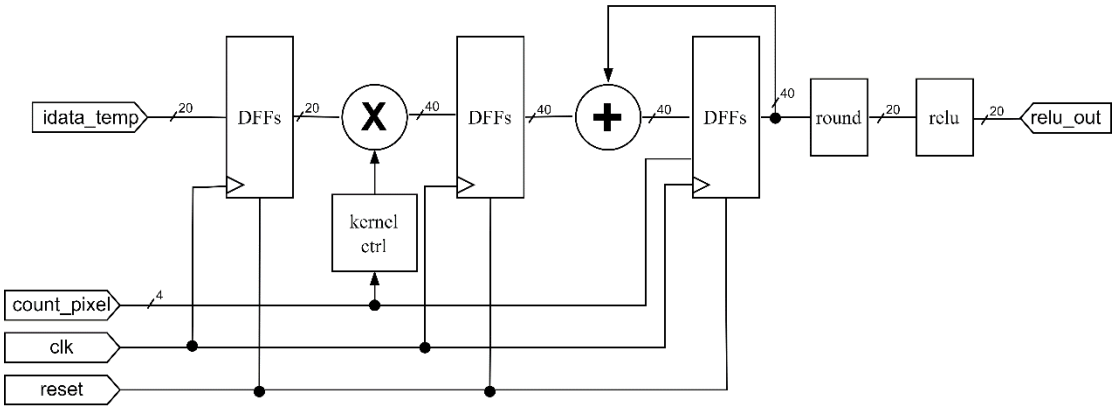


▲圖二 b

此模塊控制輸出 cdata_wr 為 L0（relu_out）還是 L1（max）的運算數值，我們一樣使用 count_pixel 控制多工器，先經過暫存再輸出。

（四）Computation L0 / L1：

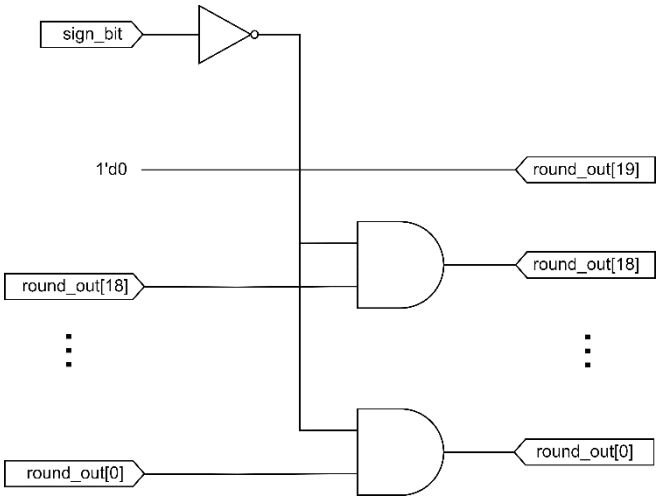
1. L0 計算：



▲圖二 c-1

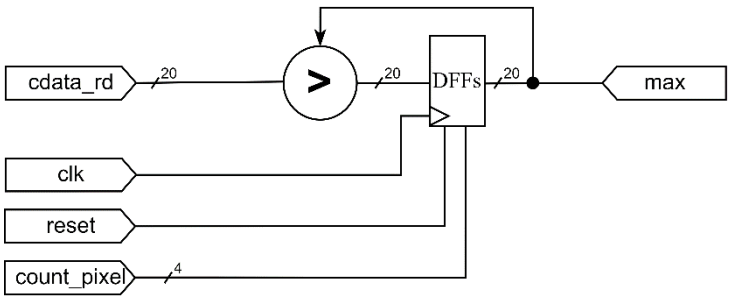
此為 L0 的卷積運算模塊，這裡採 pipeline 設計方法，輸入像素存入 DFF 再和 kernel 乘法運算完成之後，以 DFF 暫存，接著再進行加法，再輸入暫存，再進行四捨五入和 relu。而我們使用 count_pixel 決定何時 DFF 需要重置，並且決定 kernel 的數值。

這部分我們 rounding 方式是以擷取位元方式進行，relu 方式如以下電路架構，以 gate level 完成。



▲圖二 c-2

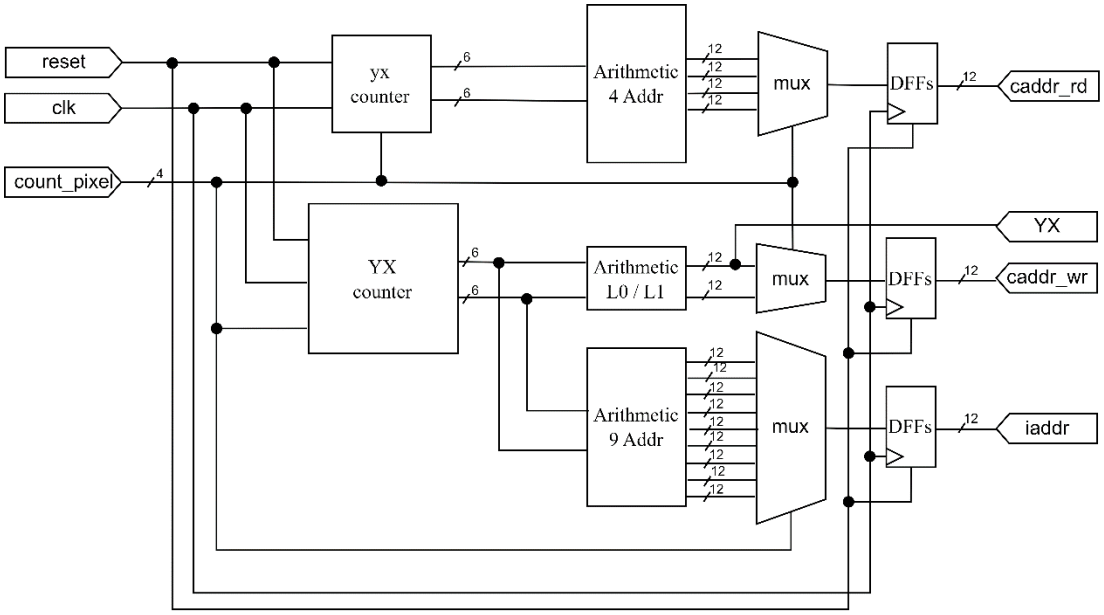
2. L1 計算：



▲圖二 d

此為 L1 的卷積運算模塊，設計方法一樣是採 pipeline 設計，依次讀取一個 L0 memory 數值 cdata_rd，再一次次比較像素大小，輸出比較最大數值 max。

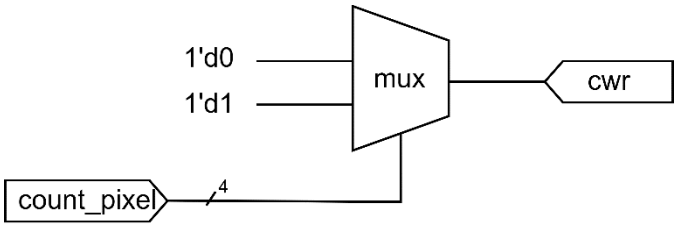
（五）Addr generator（Address generator）：



▲圖二 e

此模塊控制讀取和寫入記憶體的位址，這裡採用 YXcounter 定位 0~4095 的記憶體位址，而 yxcounter 用來決定 maxpool 的讀取的記憶體之定位。以下詳述個輸出訊號之產生方式。caddr_rd：此訊號是用來決定 maxpool 的輸入像素位址，用 yxcounter 產生定位位址，再透過 Arithmetic 4 Addr 模塊產生四個像素定位的位址，再由多工器依照目前 count_pixel 數值決定輸出的位址。YX：此為 0~4095 的定位數值，輸出是為了提供其他模塊作為判斷訊號。caddr_wr：此訊號是用來決定寫入記憶體的位址，由 YXcounter 產生，再經過 Arithmetic L0 / L1 決定輸出 L0 或是 L1 的計算位址。iaddr：此訊號用來決定讀取卷積運算像素位址，由 YXcountert 產生定位，再經過 Arithmetic 9 Addr 產生 9 個位址送入 mux，經過 count_pixel 決定 mux 輸出為哪個。這些訊號輸出前皆經過一暫存器用以穩定訊號。

（六）signal ctrl（signal control）：



▲圖二 f

對應圖二中的 signal ctrl，此模塊產生 busy、cwr、crd、csel 之控制訊號，此模塊為 combinational 電路，內部架構皆相似，以 mux 決定輸出數值，控制訊號為 count_pixel。以 cwr 為例，當 count_pixel 為對應數值則輸出 0 或 1。

貳、 綜合比較：

以下我們將此兩個版本電路進行架構之評比，列出各自優缺點。

	版本一	版本二
優點	共用 1 個位址 counter 2 個 pipeline DFFs	1 個乘法器 1 個加法器
缺點	3 個乘法器 3 個加法器	3 個 pipeline DFFs 3 個位址 counter

參、 電路分析

一、 版本一電路

```
****************************************
Report : area
Design : CONV
Version: P-2019.03
Date   : Sat Jun 22 00:10:30 2024
****************************************

Information: Updating design information... (UID-85)
Library(s) Used:

    slow (File: /mnt3/CBDK_IC_Contest_v2.1/SynopsysDC/db/slow.db)

Number of ports:                         1350
Number of nets:                          7725
Number of cells:                         5849
Number of combinational cells:           5316
Number of sequential cells:               503
Number of macros/black boxes:               0
Number of buf/inv:                       1047
Number of references:                     116

Combinational area:              65085.105652
Buf/Inv area:                     7215.647428
Noncombinational area:           16760.127106
Macro/Black Box area:                0.000000
Net Interconnect area:          718763.606201

Total cell area:                 81845.232758
Total area:                     800608.838959

***** End Of Report *****
```

```
AOI222X4                slow          22.066200      29     639.919807
AOI222XL                slow          13.579200      95    1290.023980
BUFX4                   slow           8.487000       7      59.409003
BUFX12                  slow          16.974001      27     458.298025
BUFX16                  slow          23.763599       1      23.763599
BUFX20                  slow          28.855801       1      28.855801
CLKAND2X3               slow           8.487000       7      59.409003
CLKAND2X4               slow          10.184400       1      10.184400
CLKAND2X8               slow          15.276600       2      30.553200
CLKBUFX2                slow           6.789600       1       6.789600
CLKBUFX3                slow           6.789600     102     692.539189
CLKINVX1                slow           3.394800     122     414.165594
CLKINVX2                slow           5.092200       1       5.092200
CLKINVX3                slow           5.092200       1       5.092200
CLKINVX8                slow          10.184400       2      20.368799
CLKMX2X2                slow          13.579200      24     325.900795
CONV_DW01_add_2                     4095.826265       1    4095.826265  h
CONV_DW01_add_3                     3208.086033       1    3208.086033  h
CONV_DW01_add_4                     2858.421580       1    2858.421580  h
CONV_DW01_add_5                     2943.291592       1    2943.291592  h
CONV_DW_mult_tc_3                  11700.178245       1   11700.178245  h
CONV_DW_mult_tc_4                  11326.750249       1   11326.750249  h
CONV_DW_mult_tc_5                  11941.209044       1   11941.209044  h
DFFRHQX4                slow          49.224602       3     147.673805  n
DFFRHQX8                slow          54.316799       2     108.633598  n
DFFRX1                  slow          32.250599     438   14125.762321  n
DFFRX2                  slow          37.342800      39    1456.369205  n
DFFRX4                  slow          47.527199      16     760.435181  n
DFFRXL                  slow          32.250599       5     161.252995  n
INVX1                   slow           3.394800       3      10.184400
INVX2                   slow           5.092200       1       5.092200
INVX3                   slow           5.092200      23     117.120595
INVX4                   slow           6.789600       3      20.368800
INVX8                   slow          10.184400       3      30.553199
INVX12                  slow          13.579200      42     570.326391
INVX16                  slow          16.974001       3      50.922003
```

合成後面積：

可從 Total cell area 得知，邏輯閘數量約為 81845.232758/5=16369.0466 個邏輯閘

可從 Design>Report Reference 觀察哪些部分占了面積的大宗

可得知三個乘法器就佔了 34968.1375um^2

加法器和 Flip-Flop 也同樣加起來占了約為 27231.3052um^2

這些地方基本上佔了總體電路的約 80%面積，因此需要降低乘法器與加法器數量才會是最快能降低面積的優化方向。

執行時間：

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

START!!! Simulation Start .....

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

Layer 0 (Convolutional Output) with Kernel 0 is correct !
Layer 1 (Max-pooling Output) with Kernel 0 is correct!

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

-------------------- S U M M A R Y -----------------

Congratulations! Layer 0 data have been generated successfully! The result is PASS!!

Congratulations! Layer 1 data have been generated successfully! The result is PASS!!

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

Simulation complete via $finish(1) at time 399451122 PS + 0
./testfixture.v:191       #(`CYCLE/2); $finish;
ncsim> exit
Godzilla [~/112-2_PDSD_Final_Project_HW]
-DTARS112a10- $
```

可得知運算時間為 399451.122ns

代表整個狀態表中，經過了約 39945 個狀態，因此如果要優化時間的話就要思考要如何減少狀態數去下手。

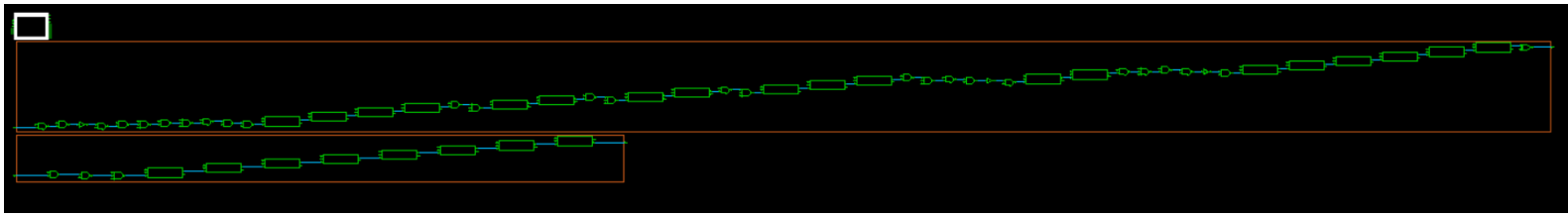
功耗：

```
****************************************
Report : power
         -analysis_effort low
Design : CONV
Version: P-2019.03
Date   : Sat Jun 22 00:32:14 2024
****************************************


Library(s) Used:

    slow (File: /mnt3/CBDK_IC_Contest_v2.1/SynopsysDC/db/slow.db)


Operating Conditions: slow   Library: slow
Wire Load Model Mode: top

Design        Wire Load Model            Library
------------------------------------------------
CONV                  tsmc13_wl10       slow


Global Operating Voltage = 1.08
Power-specific unit information :
    Voltage Units = 1V
    Capacitance Units = 1.000000pf
    Time Units = 1ns
    Dynamic Power Units = 1mW    (derived from V,C,T units)
    Leakage Power Units = 1pW


   Cell Internal Power  =   1.2880 mW   (72%)
   Net Switching Power  = 502.2939 uW   (28%)
                          ---------
Total Dynamic Power     =   1.7903 mW  (100%)

Cell Leakage Power      =  76.5769 uW
```

可得知動態功耗為 1.7903mW

如同上課教授所說，面積通常與功耗會成正相關，因此若要降低功耗可以試著從縮面積下手。

最長路徑：



最長路徑有兩條、皆為加法器內部的架構

(從 B[9]至 sum[39]與從 A[0]至 sum[9])

因為我們是利用合成軟體所提供的加法器利用 adder tree 進行實作，因此要優化的話可以試著從寫一個最長路徑短一點的加法器下手。

```
data arrival time                                                  10.16

clock clk (rise edge)                                 10.00        10.00
clock network delay (ideal)                            0.50        10.50
clock uncertainty                                     -0.10        10.40
part_add0_reg[39]/CK (DFFRX1)                          0.00        10.40 r
library setup time                                    -0.23        10.17
data required time                                                 10.17
-----------------------------------------------------------------------------
data required time                                                 10.17
data arrival time                                                 -10.16
-----------------------------------------------------------------------------
slack (MET)                                                        0.00
```

```
Startpoint: counter_X_reg[3]
            (rising edge-triggered flip-flop clocked by clk)
Endpoint: caddr_wr_reg[3]
          (rising edge-triggered flip-flop clocked by clk)
Path Group: clk
Path Type: min

Des/Clust/Port     Wire Load Model       Library
------------------------------------------------
CONV               tsmc13_wl10           slow

Point                                    Incr       Path
-----------------------------------------------------------
clock clk (rise edge)                    0.00       0.00
clock network delay (ideal)              0.50       0.50
counter_X_reg[3]/CK (DFFRX4)             0.00       0.50 r
counter_X_reg[3]/Q (DFFRX4)              0.40       0.90 r
caddr_wr_reg[3]/D (DFFRX1)               0.00       0.90 r
data arrival time                                   0.90

clock clk (rise edge)                    0.00       0.00
clock network delay (ideal)              0.50       0.50
clock uncertainty                        0.10       0.60
caddr_wr_reg[3]/CK (DFFRX1)              0.00       0.60 r
library hold time                       -0.12       0.48
data required time                                  0.48
-----------------------------------------------------------
data required time                                  0.48
data arrival time                                  -0.90
-----------------------------------------------------------
slack (MET)                                        0.42
```

可從上面兩圖中看到，此電路 slack 皆為正的，並無違反 timing violation

(左圖為 delay type 設 max、右圖為 delay type 設 min)

若想要 slack 變大，可以試著縮短最長路徑下手，也就是自己寫一個加法器或改架構來減少加法器串接的數量。

二、 版本二電路

合成後面積：

```
****************************************
Report : area
Design : CONV
Version: P-2019.03
Date   : Fri Jun 21 09:56:03 2024
****************************************

Library(s) Used:

    slow (File: /mnt3/CBDK_IC_Contest_v2.1/SynopsysDC/db/slow.db)

Number of ports:                          394
Number of nets:                          2222
Number of cells:                         1673
Number of combinational cells:           1457
Number of sequential cells:               208
Number of macros/black boxes:               0
Number of buf/inv:                        268
Number of references:                      64

Combinational area:              18875.087957
Buf/Inv area:                     1948.615183
Noncombinational area:            6792.994579
Macro/Black Box area:                0.000000
Net Interconnect area:          199221.003723

Total cell area:                 25668.082536
Total area:                     224889.086259

***** End Of Report *****
```

可從 Total cell area 得知，邏輯閘數量約為 25668.082536/5=5133.6165 個邏輯閘

Reference	Library	Unit Area	Count	Total Area	A
ADDHXL	slow	18.671400	8	149.371201	r
AND2X2	slow	6.789600	7	47.527199	
AND2X4	slow	10.184400	1	10.184400	
AND2XL	slow	6.789600	2	13.579200	
AND3X1	slow	8.487000	1	8.487000	
AND3X2	slow	8.487000	4	33.948002	
AO21XL	slow	8.487000	2	16.974001	
AOI2BB1X1	slow	8.487000	1	8.487000	
AOI21X1	slow	8.487000	1	8.487000	
BUFX4	slow	8.487000	1	8.487000	
BUFX12	slow	16.974001	1	16.974001	
CLKAND2X8	slow	15.276600	1	15.276600	
CLKBUFX3	slow	6.789600	32	217.267197	
CLKINVX1	slow	3.394800	22	74.685599	
CLKINVX2	slow	5.092200	1	5.092200	
CLKINVX6	slow	8.487000	2	16.974001	
CONV_DW01_add_1		3377.826038	1	3377.826038	h
CONV_DW01_inc_1		366.638401	1	366.638401	h
CONV_DW_cmp_1		566.931598	1	566.931598	h
CONV_DW_mult_tc_2		10452.589192	1	10452.589192	h
DFFRX1	slow	32.250599	189	6095.363194	n
DFFRX2	slow	37.342800	7	261.399601	n
DFFRX4	slow	47.527199	6	285.163193	n
DFFSX1	slow	30.553200	4	122.212799	n
INVX1	slow	3.394800	3	10.184400	
INVX3	slow	5.092200	6	30.553199	
INVX12	slow	13.579200	60	814.751987	
INVXL	slow	3.394800	2	6.789600	
NAND2BX1	slow	6.789600	6	40.737599	
NAND2BXL	slow	6.789600	1	6.789600	
NAND2X1	slow	5.092200	13	66.198597	
NAND2X2	slow	8.487000	4	33.948002	
NAND2X4	slow	11.881800	1	11.881800	
NAND2XL	slow	5.092200	4	20.368799	
NAND3BX1	slow	8.487000	3	25.461001	

可從 Design>Report Reference 觀察哪些部分占了面積的大宗

可得知乘法器就佔了 10452.6um^2

加法器和 Flip-Flop 也同樣加起來占了約為 9473.18um^2

這些地方基本上佔了總體電路的 80%面積，因此是個可優化的下手目標。

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

START!!! Simulation Start .....

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

Layer 0 (Convolutional Output) with Kernel 0 is correct !
Layer 1 (Max-pooling Output) with Kernel 0 is correct!

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

--------------------- S U M M A R Y -----------------

Congratulations! Layer 0 data have been generated successfully! The result is PASS!!

Congratulations! Layer 1 data have been generated successfully! The result is PASS!!

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

Simulation complete via $finish(1) at time 532561558 PS + 0
./testfixture.v:191       #(`CYCLE/2); $finish;
ncsim> exit
Godzilla [~/112-2_PDSD_Final_Project_HW]
-DTARS112a10- $
```

執行時間：

可得知運算時間為 532561.558ns

其實可以從演算法當中推算，總體運算時間會約為

13(每個 pixel 為捲積中心的運算 clock 數)*4096(64*64 格)*10(clock period)

因此約為 532480ns 附近，與實測結果相近，差異在於最後要把 write_down 拉起來會有幾個 buffer 以及現實的邏輯閘延遲導致了現在的結果，因此如果要優化時間的話就要思考要如何減少每個 pixel 為捲積中心的運算 clock 數去下手。

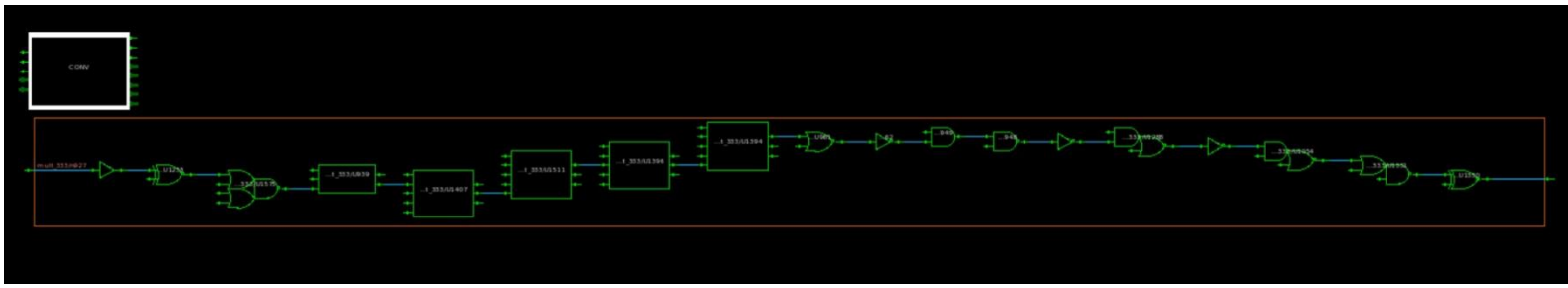
功耗：

```
****************************************
Report : power
        -analysis_effort low
Design : CONV
Version: P-2019.03
Date   : Fri Jun 21 09:56:31 2024
****************************************


Library(s) Used:

    slow (File: /mnt3/CBDK_IC_Contest_v2.1/SynopsysDC/db/slow.db)


Operating Conditions: slow   Library: slow
Wire Load Model Mode: top

Design        Wire Load Model            Library
------------------------------------------------
CONV                  tsmc13_wl10       slow


Global Operating Voltage = 1.08
Power-specific unit information :
    Voltage Units = 1V
    Capacitance Units = 1.000000pf
    Time Units = 1ns
    Dynamic Power Units = 1mW    (derived from V,C,T units)
    Leakage Power Units = 1pW


   Cell Internal Power  = 552.4386 uW   (53%)
   Net Switching Power  = 494.9614 uW   (47%)
                          ---------
Total Dynamic Power     =   1.0474 mW  (100%)

Cell Leakage Power      =  26.3794 uW
```

可得知動態功耗為 1.0474mW

如同上課教授所說，面積通常與功耗會成正相關，因此若要降低功耗可以試著從縮面積下手。

最長路徑：



最長路徑為乘法器內部的架構

因為我們是利用合成軟體所提供的乘法器進行實作，因此要優化的話可以

```
data arrival time                                      10.12

clock clk (rise edge)                      10.00       10.00
clock network delay (ideal)                 0.50       10.50
clock uncertainty                          -0.10       10.40
mul_reg[31]/CK (DFFRX1)                     0.00       10.40 r
library setup time                         -0.27       10.13
data required time                                     10.13
-----------------------------------------------------------------------------
data required time                                     10.13
data arrival time                                     -10.12
-----------------------------------------------------------------------------
slack (MET)                                            0.00
```

```
Operating Conditions: slow   Library: slow
Wire Load Model Mode: top

  Startpoint: caddr_wr_temp_temp_reg[0]
              (rising edge-triggered flip-flop clocked by clk)
  Endpoint: caddr_wr_reg[0]
            (rising edge-triggered flip-flop clocked by clk)
  Path Group: clk
  Path Type: min

  Des/Clust/Port     Wire Load Model       Library
  ------------------------------------------------
  CONV               tsmc13_wl10           slow

  Point                                                   Incr       Path
  ------------------------------------------------------------------------------
  clock clk (rise edge)                                   0.00       0.00
  clock network delay (ideal)                             0.50       0.50
  caddr_wr_temp_temp_reg[0]/CK (DFFRX1)                   0.00       0.50 r
  caddr_wr_temp_temp_reg[0]/Q (DFFRX1)                    0.47       0.97 r
  caddr_wr_reg[0]/D (DFFRX1)                              0.00       0.97 r
  data arrival time                                                  0.97

  clock clk (rise edge)                                   0.00       0.00
  clock network delay (ideal)                             0.50       0.50
  clock uncertainty                                       0.10       0.60
  caddr_wr_reg[0]/CK (DFFRX1)                             0.00       0.60 r
  library hold time                                      -0.13       0.47
  data required time                                                 0.47
  ------------------------------------------------------------------------------
  data required time                                                 0.47
  data arrival time                                                 -0.97
  ------------------------------------------------------------------------------
  slack (MET)                                                        0.50
```

試著從寫一個最長路徑短一點的乘法器下手。

可從上面兩圖中看到，此電路 slack 皆為正的，並無違反 timing violation

(左圖為 delay type 設 max、右圖為 delay type 設 min)

若想要 slack 變大，可以試著縮短最長路徑下手，也就是自己寫一個乘法器。

三、 綜合比較

版本\效能	面積(um^2)	執行時間(ns)	功耗(mW)	效能(Area*Timing，um^2*ns)
版本一電路	81845.232758	399451.122	1.7903	32,693,170,055.53425
版本二電路	25668.082536	532561.558	1.0474	13,669,834,026.24475

從表格得知，版本一電路執行速度比版本二快約 25%，但面積是版本二的 3.2 倍。根據效能公式，版本一的總體效能為版本二的 2.4 倍。版本一需要 3 個乘法器和 4 個加法器，使其面積達到版本二的兩倍，且難以壓縮執行時間至版本二的兩倍內。最終選擇使用 1 個乘法器和 1 個加法器的版本二，降低了面積，稍微增加執行時間，但大幅提升效能，功耗也更低，顯示版本二的演算法及電路設計更優。