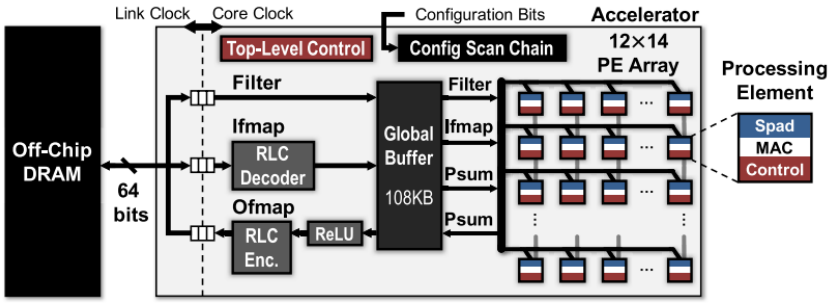
# AOC Paper Reading and Review 3
# Eyeriss: An Energy-Efficient Reconfigurable Accelerator for Deep
# N26122246 胡家豪

## 1. Motivation

本篇 paper 提出了一種 CNN 的加速器，稱為「Eyeriss」，其由 168 個 PE 組成，並且作者提倡使用「Row stationary」的 data flow，並且使用了「RLE」的方式將數據進行壓縮。在這個架構之下，作者表示「Eyeriss」能夠達到 high throughput 與 energy-efficiency

## 2. Proposed solution

一.Eyeriss Architecture

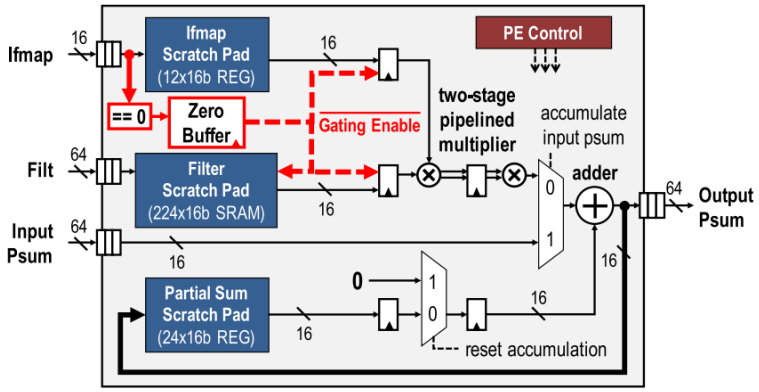



在本篇的 Eyeriss 架構中，分別有 Core Clock 與 Link Clock 兩種 Clock，其透過 Asynchronous FIFO 將 DRAM 內的 ifmap、Filter 傳入。

1.Top-Level Control

Top Control 主要負責 DRAM、Global Buffer 以及 PE Array 之間的溝通。其控制 DRAM 取出 Filter、Ifmap 至 chip 上的 Buffer、控制 Buffer 取出 Filter、Ifmap 至 PE array 與 Psum 的累加。

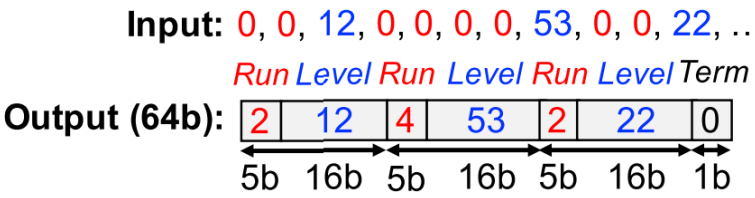
2.lower -Level Control




在每一個 PE 單元內均有一個 control，並且彼此之間互相獨立。

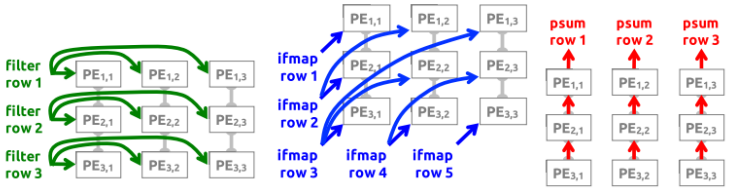
Lower Control 控制一般 PE 的 MAC 運算，值得一提的是作者在 ifmap 中有額外進行一個判斷是否為 0 的邏輯，如果成立便會跳過後面的 dataflow 直接執行下一個非 0 的運算，如此一來可以減少 45%的功耗。

二. RLE(Run Length Encoding)




由於作者發現較知名且有效的模型，通常會進行 ReLU 運算。如此一來會導致 ofmap 的結果有很多 0 的出現，所以作者在 ofmap 進行 RLE，如此一來可以用較少的 bits 表示 0 的數量，達到節省存儲空間的目的。

三. Row stationary




Row stationary 是本篇首次提出的 data flow，主要作法是將 Filter 與 ifmap 以 row 為單位進行運算。
如此是為了最大化減少各種資料的 data movement(不論是 ifmap, filter 或 psum)，以達到 energy efficienct 的效果。

# 3. Evaluation

1.Performance

作者使用「Eyeriss」執行 VGG-16 的 model，Batch size 設為 3，Supply voltage 為 1V，on-Chip clock 為 200MHz ：外部的 clock 為 60MHz。

| Layer | Power (mW) | Total Latency (ms) | Processing Latency (ms) | Num. of MACs | Num. of Active PEs | Zeros in Ifmaps (%) | Global Buff. Accesses | DRAM Accesses |
|---|---|---|---|---|---|---|---|---|
| CONV1-1 | 247 | 76.2 | 38.0 | 0.26G | 156 (93%) | 1.6% | 112.6 MB | 15.4 MB |
| CONV1-2 | 218 | 910.3 | 810.6 | 5.55G | 156 (93%) | 47.7% | 2402.8 MB | 54.0 MB |
| CONV2-1 | 242 | 470.3 | 405.3 | 2.77G | 156 (93%) | 24.8% | 1201.4 MB | 33.4 MB |
| CONV2-2 | 231 | 894.3 | 810.8 | 5.55G | 156 (93%) | 38.7% | 2402.8 MB | 48.5 MB |
| CONV3-1 | 254 | 241.1 | 204.0 | 2.77G | 156 (93%) | 39.7% | 607.4 MB | 20.2 MB |
| CONV3-2 | 235 | 460.9 | 408.1 | 5.55G | 156 (93%) | 58.1% | 1214.8 MB | 32.2 MB |
| CONV3-3 | 233 | 457.7 | 408.1 | 5.55G | 156 (93%) | 58.7% | 1214.8 MB | 30.8 MB |
| CONV4-1 | 278 | 135.8 | 105.1 | 2.77G | 168 (100%) | 64.3% | 321.8 MB | 17.8 MB |
| CONV4-2 | 261 | 254.8 | 210.0 | 5.55G | 168 (100%) | 74.7% | 643.7 MB | 28.6 MB |
| CONV4-3 | 240 | 246.3 | 210.0 | 5.55G | 168 (100%) | 85.4% | 643.7 MB | 22.8 MB |
| CONV5-1 | 258 | 54.3 | 48.3 | 1.39G | 168 (100%) | 79.4% | 90.0 MB | 6.3 MB |
| CONV5-2 | 236 | 53.7 | 48.5 | 1.39G | 168 (100%) | 87.4% | 90.0 MB | 5.7 MB |
| CONV5-3 | 230 | 53.7 | 48.5 | 1.39G | 168 (100%) | 88.5% | 90.0 MB | 5.6 MB |
| **Total** | **236** | **4309.5** | **3755.2** | **46.04G** | **158 (94%)** | **58.6%** | **11035.8 MB** | **321.1 MB** |

Power(mW)：每一層所花的功率，(total 為平均功率)
Total Latency (ms)：完成此層所花費的時間
Processing Latency(ms)：單純進行 MAC 所花費的時間
Num of MACs：此層所執行的乘加運算數目
Num of Active PE：在執行此層運算，work 的 PE 比例
Zero in Ifmaps：此層運算時，input feature 有 0 的比率
Global Buff Accesses：執行此層運算時，on Chip 的 Buffer 存取了多少 data
DRAM Accesses：執行此層運算時，DRAM 的 Buffer 存取了多少 data

## 2. My analysis

我認為 Eyeriss 提出的「Row stationary」算是一個蠻大的創舉。在他之前多數的架構都是用「weight stationary」或是「output stationary」等等比較傳統的方式將一種資料固定，對其進行重複利用。

而 Eyeriss 提出的「Row stationary」則是首次將每一個 ROW 一次計算完畢以後再進行下一個 ROW 的計算，如此可以最大化 weight 和 activation 在 PE array 的使用次數。如此一來就可以降低功耗。

但是 Eyeriss 的 PE 相較於一般的 systolic MAC，Eyeriss 的 PE 控制更加複雜、而且由於 PE 內部需要 SPAD，所以面積會比較大。

而且由於 PE 需要等待 ifmap 與 filter 的 procast，比起 systolic 的計算還是較為小一些，所以 Eyeriss 還是在能耗方面比較有優勢。

不過在讀這篇 paper 之前，過去我只知道 systolic array 它的核心精神是將資料盡可能的併行處理與讓資料在 PE 單元內多「流動」，以此減少記憶體的存取，達到加速與節能。但是對於各種 stationary 的方法雖然聽過，但是都不太知道具體是如何操作，這篇論文讓我更加了解了加速器內部的結構與操作。