NYCU-ECE DCS-2023

Final Project

Design: DCSformer

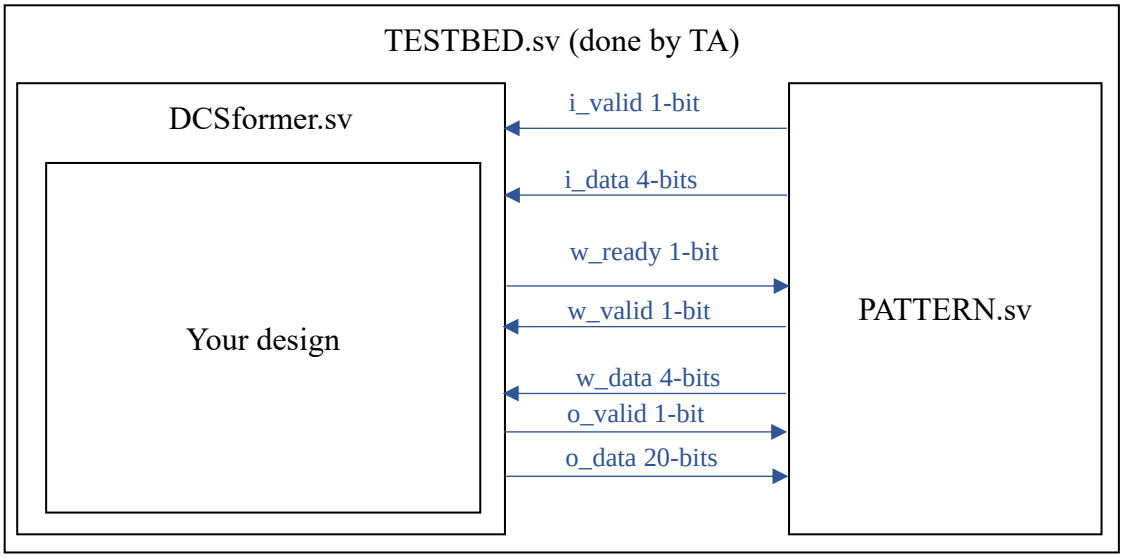
Preparation

1. 從 TA 目錄資料夾解壓縮:

   % tar -xvf ~dcsTA01/Final.tar

2. 解壓縮資料夾 Final 包含以下:
   a. 00_TESTBED/
   b. 01_RTL/
   c. 02_SYN/
   d. 03_GATE/

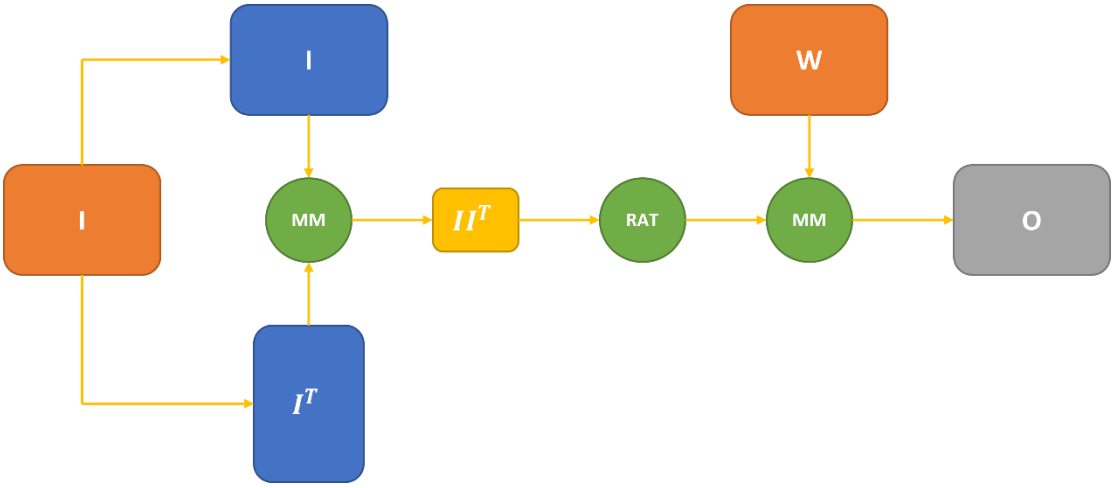
Block Diagram



這次Final Project可以自己調整clock period ！詳情請看Specification。

Transformer在近幾年的AI領域中可說是最具影響力的架構，大家耳熟能詳的ChatGPT便是基於此架構下的強大模型。不只是生成式AI領域，影像處理以及語音處理等，基於Transformer衍生出的各種Xformer架構更是多次刷新了SOTA。這次作業主要是針對DCSformer中的Attention Block之計算。

其運算方式如下圖所示：



- I, W分別為 input matrix 以及 weight matrix
- O為最後的輸出
- MM代表者矩陣-矩陣(向量)的乘法運算
- RAT, Row-wise Average Threshold為一個激活函數，針對矩陣中的每個列(row)進行各別運算，其運算規則為：
  - 找出改列的平均值(這邊的平均求法，**為先求總和，再除以個數；倘若同學個別除以總合，再求總和，需特別注意**rounding的問題)，並以此為閥值。
  - 當列中的數值大於或等於此閥值，則保留該值；當小於閥值時，則改為零。
  - 其pseudo code為：

```
def RAT(i_mtx):
    T1, T2 = i_mtx.size()
    for i in range(T1):
        threshold = avg(i_mtx[i])
        for j in range(T2):
            if i_mtx[i][j] < threshold:
                o_mtx[i][j] = 0
            else:
                o_mtx[i][j] = i_mtx[i][j]
    return o_mtx
```

- 輸入矩陣(I)與其轉置矩陣(I^T)做矩陣-矩陣乘法，接下來執行RAT之激活函數。然後將結果與權重矩陣(W)做矩陣-向量乘法，即可輸出。

以下為本次作業之介紹：

- Input

有兩種不同類別的輸入模式，分別是輸入input模式和輸入weight模式。

- 輸入input模式

當i_valid拉起時代表pattern向design輸入input matrix的數值，此input為一個8×16的矩陣，pattern會以以raster scan order的方式輸入，持續128個時間週期。數值大小範圍為0~255。

- 輸入weight模式

當w_valid拉起時代表pattern向design輸入weight matrix的數值，此weight為一個8×1的向量，pattern會以以raster scan order的方式輸入，持續8個時間週期。數值大小範圍為0~255。

- Output

運算完後的結果以raster scan order方式輸出，一次輸出一個element，總共輸出8個時間週期。注意：**o_valid訊號不能與w_valid訊號重疊**；在reset結束後，尚未輸出資料時，**o_valid和o_data須維持為low**，不可出現unknown, high-Z, 或任何除了零以外的值。

- Pattern和Design的互動模式

首先會先進入輸入input模式，待你的design做完前處理的運算，且準備要處理線性轉換之運算時，將w_ready拉起一個cycle以告知pattern，你的design已經準備好接收若干筆weight資料。w_ready拉起後的2~6個negedge clock cycle後，會進入輸入weight模式，待完成所有運算後，將o_valid拉起，依序輸出o_data。在間隔5~10個negedge clock cycle後，pattern會輸入下一筆測資。

總而言之，pattern和design會反覆執行以下流程：

1. i_valid拉起且輸送i_data至design →
2. design做前處理之運算 →
3. design以w_ready與pattern溝通 →
4. w_valid拉起且輸送w_data至design →

5. design做線性轉換，並輸出o_valid及o_data以供pattern檢查 →
6. pattern提供下一筆測資

一些訊號的注意事項：

- **i_valid和w_ready不可重疊**
- **o_valid和w_valid不可重疊**
- **w_ready僅能拉起 1 個時間週期**
- **o_valid需要拉起 8 個時間週期**

- Pattern

請注意，這次final project不會提供pattern，助教會在兩周後提供一個pattern範本(pattern數較少且不會涵蓋所有可能)。同學可利用前面一周的時間練習寫pattern。為了更有效率的寫pattern，助教會建議以read file的方式輸入資料及比對答案，用python或C++生成測資應該會簡單許多。

Inputs

Signal name	Number of bit	Description
clk	1-bit	Clock
rst_n	1-bit	Asynchronous active-low reset
i_valid	1-bit	拉起時代表此段時間輸入 input matrix 數值，會連續拉起 128 cycles。
in_data	8-bit	無號數，在 i_valid 拉起時以 raster scan order 方式連續輸入 input matrix 數值，持續 128 cycles
w_valid	1-bit	拉起 w_valid 時代表此段時間輸入 weight matrix 數值，會連續拉起 8 cycles。
w_data	8-bit	無號數，在 w_valid 拉起時以 raster scan order 方式連續輸入 weight vector 數值，持續 8 cycles

Outputs

Signal name	Number of bit	Description
w_ready	1-bit	當 design 將 w_ready 傳回 pattern 時，pattern 會在 2~6 cycles 後拉起 w_valid，此訊號的拉起時間只能維持一個週期。並且在每筆測資中，此訊號只能被拉起一次。
o_valid	1-bit	拉起時輸出結果，持續 8 cycles。
o_data	32-bit	無號數，在 out_valid 拉起時以 raster scan order 連續輸出結果，持續 8 cycles。

Specifications

1. Top module name:**DCSformer** (File name : **DCSformer.sv**)
2. 在非同步負準位 reset 後，所有的 output 訊號必須全部歸零。
3. 在 i_valid 結束前，不可將 w_ready 拉起，此兩訊號不可重疊。
4. Output 要在 input 結束 i_valid 歸零後的 10000 cycles 內輸出。
5. 在 w_valid 結束前，不可將 o_valid 拉起，此兩訊號不可重疊。
6. Output 要連續輸出 8 cycles，不能多不能少，且輸出的答案要正確。
7. 當 o_vallid 不為 1 時，o_data 需要全部歸零。
8. Input delay = 0.5 * clock period; Output delay = 0.5 * clock period
9. 同學們可以自行條整 clock period，但是 clock period **不得超過 20ns**。
10. 02_SYN result 不行有 error、 不能有任何 latch、不可以 timing violation。
11. 03_GATE 不能有任何 timing violation。
12. 03_GATE 的 Latency 需要要與 01_RTL 相同。
13. Coding style 和 for loop 的限制解除。但是大家需要特別注意 for loop 的使用方式，否則合成時間會太久或面積會太大。
14. **若合成時間超過 2.5 小時，則視為 fail。**

Upload Files

1. 請將Final/01_RTL裡的Conv.sv依以下命名規則重新命名後上傳至E3。
   命名規則：DCSformer_{clock cycle time}_dcsxxx.sv，xxx為工作站帳號號碼, clock cycle time請取到小數第一位。命名錯誤扣5分
2. report_dcsxxx.pdf, xxx is your server account. 上傳至E3。
3. Deadline:
   1 demo: 6/13 23:59
   2 demo: 6/20 23:59 分數7折

Note

Template folders and reference commands:

1. 01_RTL/ (RTL simulation) → **./01_run**
2. 02_SYN/ (synthesis) → **./01_run_dc**
3. 03_GATE/ (gate-level simulation) → **./01_run**

Grading policy

1. Pass the RTL& Synthesis simulation. 60%
2. Performance 30%
   - Performance = Area * Computation time
   - Computation time
     = (i_valid開始為1到o_valid結束回到0所需cycle數) * clock cycle time
3. Report 10%

   報告不超過兩頁A4，並包括以下內容：

   I. 清楚描述關於你的矩陣乘法之**dataflow**，用了多少個**乘加器**。為了自己的權益，請務必清楚描述矩陣乘法的部分，倘若有抄襲疑慮，這會是一個重要的評斷標準之一。

   II. 描述你的設計方法，包含但不限於如何加速(減少critical path)或降低面積。

   III. 基於以上，畫出你的架構圖(Block diagram)

   IV. 心得報告，不侷限於此次作業，對於作業或上課內容都可以寫下。

   V. 遇到的困難與如何解決。
4. Dofuramingo 祝大家期末順利!!!