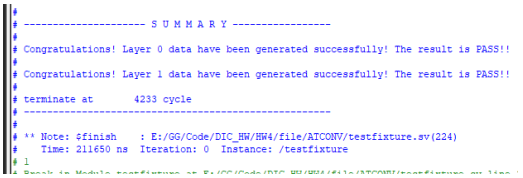
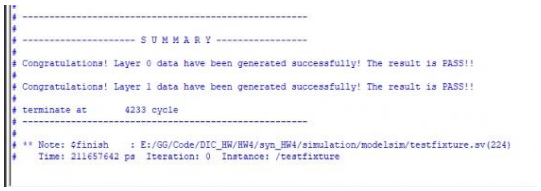
2025 Digital IC Design Homework 4

NAME	黃啟桓
Student ID	P76134927

ATCONV Simulation Result

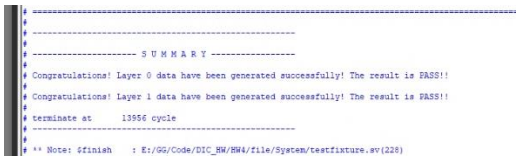
Functional simulation	Pass	Pre-Layout simulation	Pass





System Simulation Result

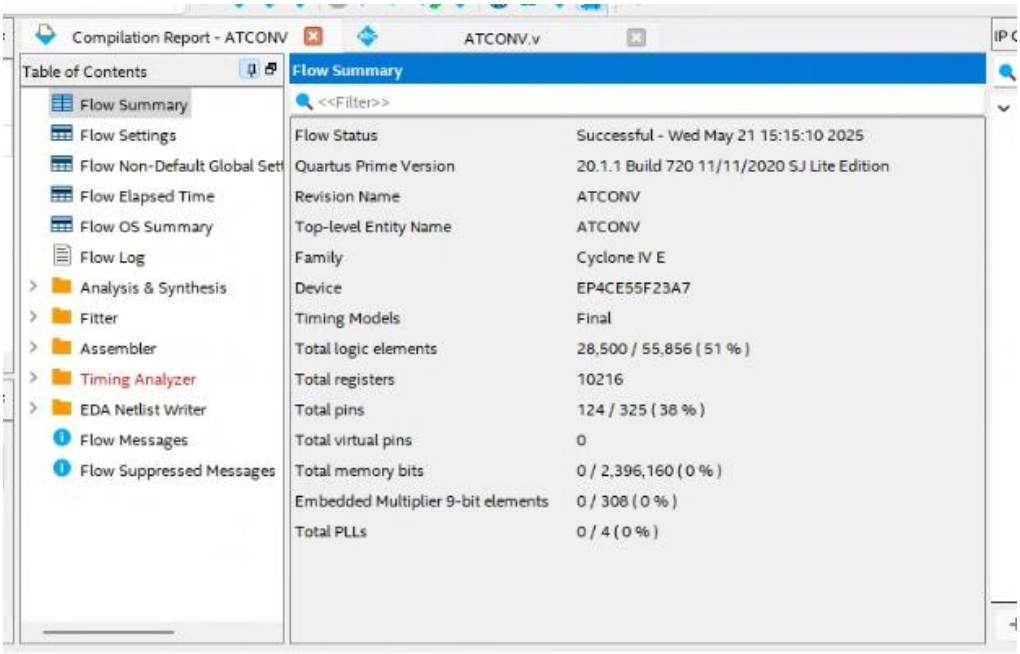
Functional simulation	Pass	Pre-Layout simulation	Fail



沒結果

ATCONV Synthesis Result

Total logic elements	28500
Total memory bits	0
Total registers	10216
Embedded multiplier 9-bit elements	0
Total Cycle used	4233 //合成需要合半小時，執行需要半小時



System Synthesis Result	
Total logic elements	無
Total memory bits	無
Total registers	無
Embedded multiplier 9-bit elements	無
Total Cycle used	無

無

Description of your design

當初就是朝著"如何做得更快"的想法去實現，所以特別壓縮計算 Layer0、Layer1 的時。
輸入的頻率最快就是 1 個 cycle，所以 output 也不用去增加 pipeline 去多組同步運算，反而是當 layer0 1 個 cycle 計算 1 個值比 1 個 cycle 計算多個值更優，因為 Layer 0 可以更快開始算，更快的輸出。Layer 1 可以更快的開始算更快的輸出。
時間上: read_raw = 4096 cycle, calc_L0 = 4096 cycle, calc_L1 = 1024 cycle, start_calc_L0 = 130 cycle(in read_raw)
已知 read_raw, calc_L0, calc_L1 可以同時進行
所以時間上最佳結果就是 4096 + 130 cycle (+ rst 甚麼時候開始) = 4226
==========
儲存資料的結構，類似使用 cycle queue 的方式，原本 raw array 需要開 68*68，但實在不需要，所以就用 64*64 去推論，接著每一次計算 L0 最多只需要 5 行，所以有效 raw_array 就保留 5*64，當然為了方便運算，採用 8*64，這樣就能直接用 shift，時間上來說肯定更快
每次輸入會先覆蓋掉最先前的 data 因為他們確實已經沒用了。

不過等我意識到 register array 開的大小會影響到 logic element 時，我已經來不及了，不想改了，所以就勉強用用
==========
System 的設計其實是延續原版，但不同的是 bus 最多一次一個資料傳輸，又考慮到其實計算 L0 L1 很快，所以不用在讀取 raw array 時，提前計算 L0 或 L1。
然後步驟就很清晰了
1. 讀取 raw 2 行。(這時還不能開始算 L0)
2. 讀取 raw 2 行。
3. 算 L0 2 行
4. 算 L1 2 行 (回到 state 2，但如果 raw 以讀取完就回到 state 3)