國立清華大學

系統晶片設計

SOC Design

國立清華大學

NATIONAL TSING HUA UNIVERSITY

Lab 3

系所級:電子所二年級

學號:111063548

姓名:蕭方凱

指導老師:賴瑾教授

目錄

I. Block Diagram 3
1. Dataflow 3
2. Block Diagram 4
3. Control Signals 5
II. Operation Description 6
Q1. How to receive data-in and tap parameters and place into SRAM? 6
Q2. How to access shiftram and tapRAM to do computation? 6
III. Resource Usage 9
(1) FF and LUT: 9
(2) Bram 9
(3) DSP 9
IV. Timing Report 10
V. Simulation Waveform 10
1. AXI-Lite 10
(1) Coefficient Program 10
(2) Coefficient Read back 10
2. RAM Access Control (Tap RAM) 11
(1) Write 11
(2) Read 11
3. AXI-Streaming 11
(1) Data-in stream-in 11
(2) Data-out stream-out 11
4. RAM Access Control (Data RAM) 12
(1) First data input 12
(2) Last Data Input 12
(3) Last Data Output 12
5. FSM 12

I. Block Diagram

1. Dataflow

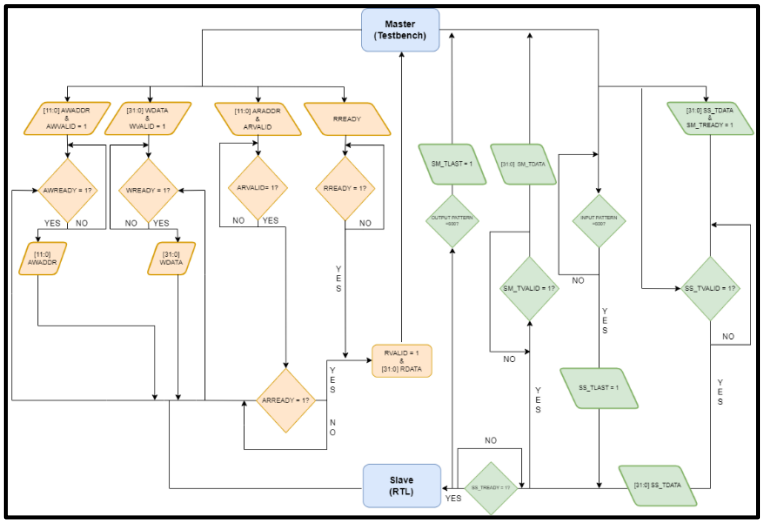


Fig 1 Dataflow Diagram

橘色部分為 axi-lite，描述 coefficient(tap)的資料流向;綠色部分為 axi-streaming，描述 streaming data(Xn)的資料流向。

(1) AXI-Lite:

Master 準備好一組 address(awaddr)並同時把 awvalid 拉至 high，告知 slave 他已準備好位址，接著等待 slave 將 awready 拉至 high 後，代表 slave 已接收到位址資料，wdata(coefficient)部分與 address 相同。完成寫入事務以後，接著由 master 準備好一組 address(araddr)並同時把 arvalid 和 rready 拉至 high，等待 slave 將 arready 拉至 high 後，當 arrvalid 及 arready 都為 high 時，rvalid 拉至 high，代表 slave 已將 master 剛剛發出的 address(araddr)對應到的 rdata(coefficient)傳回去給 master。

(2) AXI-Streaming:

Master 準備好資料流並把 ss_tvalid 拉至 1，共有 600 筆資料。等待每次 slave 將 ss_tready 拉至 1，就將資料流一筆一筆傳至 slave 直到 ss_tready 拉回 0，而 slave 會將收到的 data 拿去做 fir 運算，每算完一筆資料得到 sm_tdata 就將 sm_tvalid 拉至 1，告訴 master 已經算好此筆資料可以讀出。

2. Block Diagram

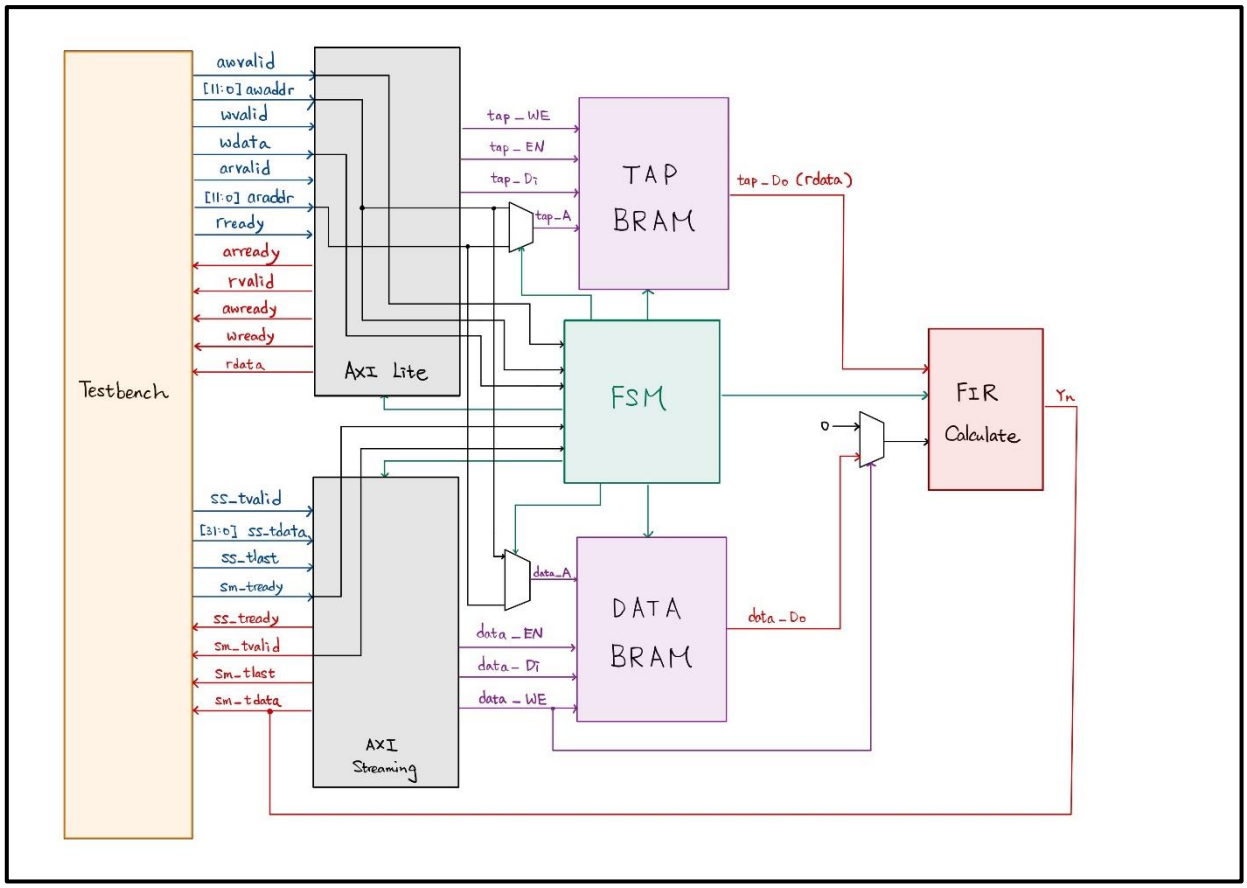


Fig 2 Block Diagram

3. Control Signals

(1) AXI-Lite:

- **Write Address Channel:**

  awready(O): Slave 已經接收到 Master 欲寫入的 address。

  awaddr(I): Master 準備寫入給 Slave 的 address。

  awvalid(I): 代表 Master 準備好一組 address 準備寫入給 Slave。

- **Write Data Channel:**

  wready(O): Slave 已經接收到 Master 欲寫入的 wdata。

  wdata(I): master 準備寫入給 slave 的 coeffiecient。

  wvalid(I): 代表 Master 準備好一組 wdata 準備寫入給 Slave。

- **Read Address Channel:**

  arready(O): Slave 已經接收到 Master 欲讀出的 address。

  araddr(I): Master 準備讀出 Slave 存放好的 address。

  arvalid(I): Master 已準備好一筆 address，告知 Slave 欲讀出的地址。

- **Read Data Channel:**

  rvalid(O): Slave 已經準備好剛剛收到的 address 對應存放的 rdata。

  rdata(O): Slave 要傳給 Master 的 coefficient。

  rready(I): Master 已經接收到 Slave 傳回的 rdata。

(2) AXI-Streaming:

- ss_tdata: Master 準備好的一組資料流 Xn。
- ss_tvalid: 代表 Master 已經準備好要給 Slave 資料 Xn。
- ss_tready: Slave 告知 Master 已準備好接收 Master 要傳的 Xn。
- ss_tlast: Master 告知 Slave 這已經是最後一筆 Xn 了
- sm_tdata: 經 FIR 算好的 Yn
- sm_tvalid: Slave 告知 Master 已準備好算好的 sm_tdata
- sm_tlast: Slave 告知 Master 這是已經是最後一筆 Yn 了
- sm_tready: Master 告知 Slave 可傳 sm_tdata

II. Operation Description

Q1. How to receive data-in and tap parameters and place into SRAM?

在 AXI-Lite 的 write 事務時，因 awvalid&wvalid 同步的，故當 awvalid 及 awready 都為 1(wvalid 及 wready 都為 1)時，將 tap_WE 拉至 1，代表可以把 address 及對應 wdata 寫入至 SRAM，也就是 tap parameters。

在 AXI-Streaming 的 write 事務時，會根據 ss_tvalid 及 sm_tvalid 的狀態去決定是否要將 data-in 傳遞給 SRAM，當 ss_tvalid&sm_tvalid 都為 1 時，就將 data_WE 拉至 1，且因為 AXI-Streaming 沒有 address 概念，所以這邊要在 RTL 部份給出對應的 address streaming(data_A)，去將 streaming data 分別對應到一個 address，如何對應哪個 address 的演算法下面會在說明。

Q2. How to access shiftram and tapRAM to do computation?

Address	**000**	**004**	**008**	**00C**	**010**	**014**	**018**	**01C**	**020**	**024**	**028**
Coefficient	**0**	**-10**	**-9**	**23**	**56**	**63**	**56**	**23**	**-9**	**-10**	**0**
Streaming Data	1	0	0	0	0	0	0	0	0	0	0
Streaming Data	1	2	0	0	0	0	0	0	0	0	0
Streaming Data	1	2	3	0	0	0	0	0	0	0	0
Streaming Data	:	:	:	:	:	:	:	:	:	:	11
Streaming Data	12	2	3	4	5	6	7	8	9	10	11
Streaming Data	12	13	3	4	5	6	7	8	9	10	11
Streaming Data	12	13	14	4	5	6	7	8	9	10	11

在寫入 tap parameters 的同時便將 data SRAM 存滿 0，完成 AXI-Lite READ 事務後，便開始存入 streaming data，存入的方式如上方表格，綠色網底代表寫入的 data 及對應到哪個 address，當 1 到 11 填滿 SRAM 時，12 便會寫入"ADDRESS 000"處，覆蓋掉 1，因為此時 1 已經不用加入 FIR 運算，而其餘沒有被覆蓋寫入的 address 其存儲的 data 則保持不變。

Tap_RAM 的部分則是一直固定給予 address 從 000 到 028 共 11 個一組的循環，其對應的 tap coefficient 也是固定的，因 tap_ram 觸發循環時機是根據 ss_tready，再加上 SRAM 的 data_Do 會比 data_Di 晚一個 clock 輸出，故 data_Do 及 tap parameters 之間會差一個 clock，透過電路設計把 streaming data 與 data_Do 經一個 delay 傳遞，兩者 clock 對齊後，即可做相乘運算。

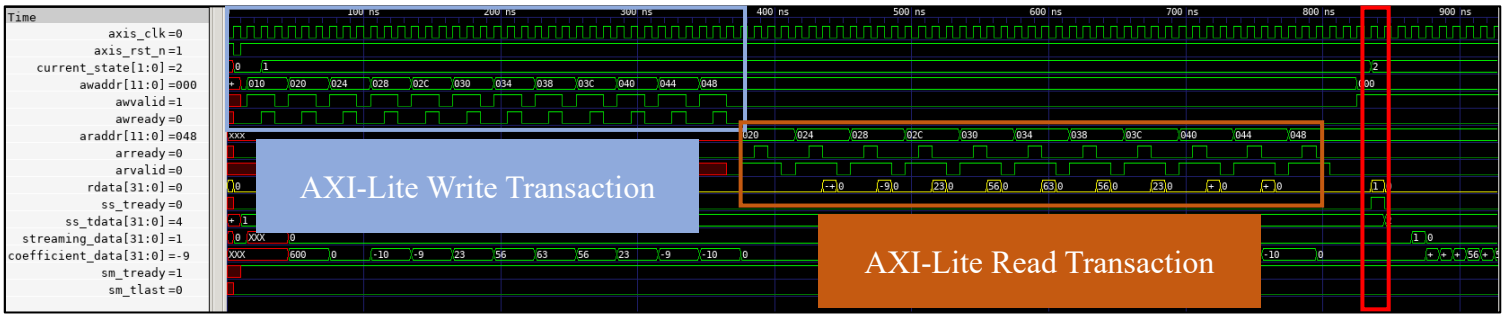
Configuration Register Address Map:

在 address = 000 的地方，其 rdata 代表的是 ap_start, ap_done, ap_idle 的狀態:

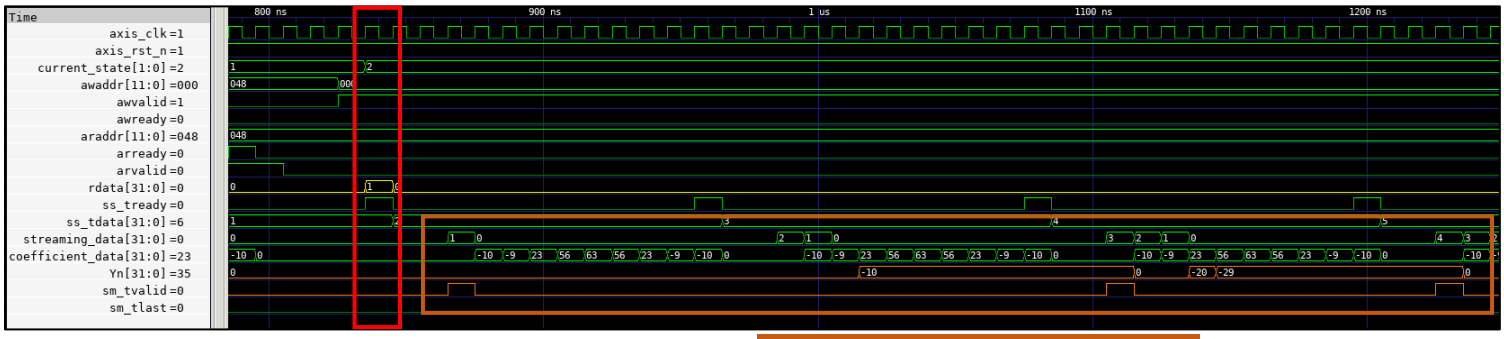
若 rdata 為 1 (ap_start)，代表第一筆 streaming data 已經寫入

若 rdata 為 2 (ap_done)，代表已完成所有 sm_tdata 的計算並輸出

若 rdata 為 4 (ap_idle)，代表回到 idle 模式，等待下一次 600 筆 pattern 輸入

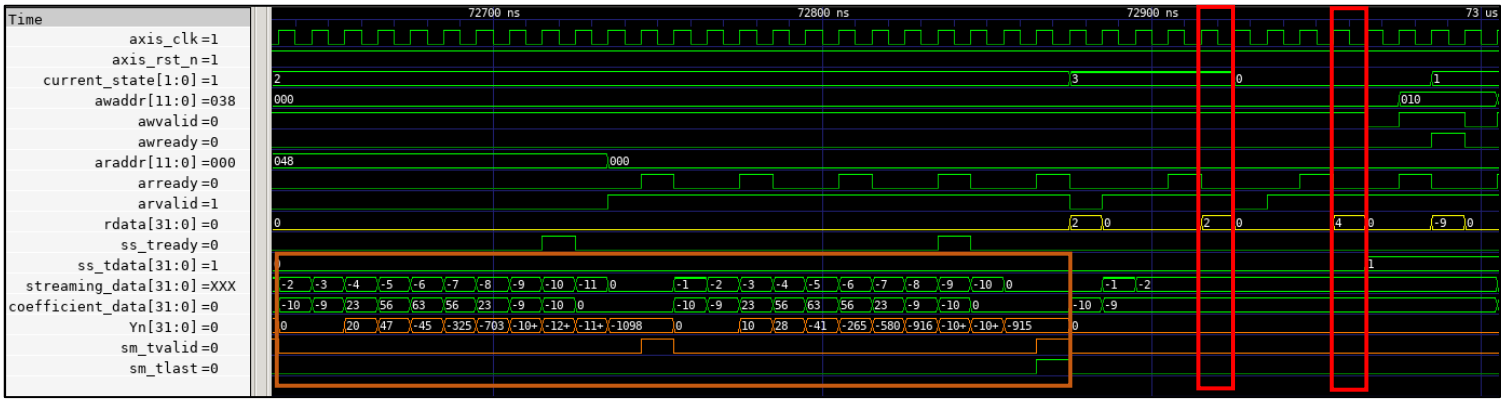


完成 AXI-Lite Read Transaction，rdata= 1



rdata= 1(ap_start)後，立即拉回 0

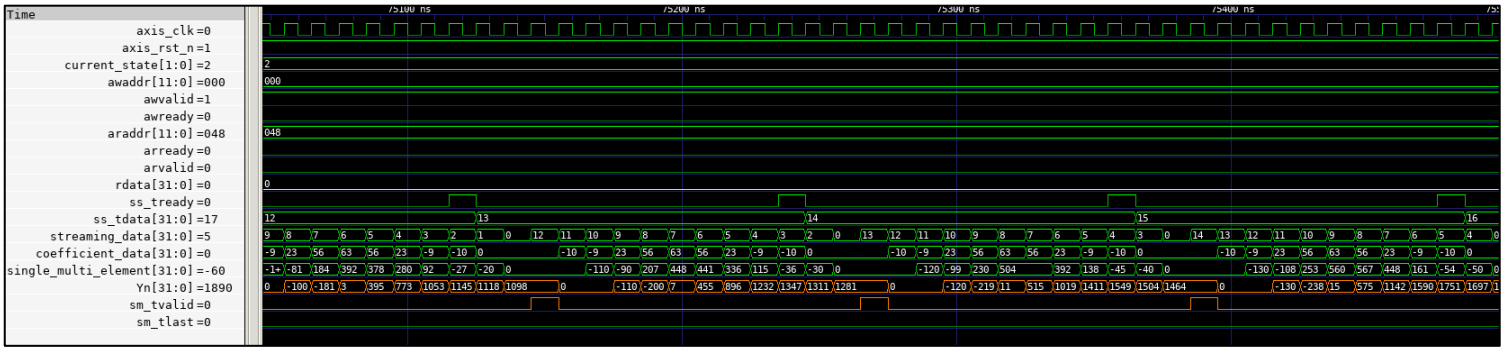
sm_tvalid = 1 時，檢查 Yn



當 sm_tlast 拉起來後，代表最後一筆 pattern 已經算完，此時 rdata 變為 2，代表 ap_done，接著 rdata 變為 4，代表回到 idle 狀態，等待新的 600 筆 pattern 輸入

rdata= 2(ap_done)並 check 完後，rdata=4(ap_idle)，回到 idle 模式

FIR Calculate:



single_multi_element 是每個 axis_clk 的 streaming data 及 coefficient data 相乘後得到的結果，而 Yn 是透過累加 single_multi_element，得到的最終輸出，會在運算完成後將 sm_tvalid 拉至 1，告知 Testbench 可以驗證答案是否正確。

FSM:

Current State	Action	Next State
IDLE	等待 tap parameters 輸入，並將所有 output signals 歸零	若輸入 tap parameters，進入 axi-lite write transaction，便進入 INPUT_COEFFICIENT，否則維持 IDLE
INPUT_COEFFICIENT	將 tap parameters 輸入給 bram，並在 testbench 想讀出時提供欲讀出之 address 對應的 rdata	若 awaddr = 000 及 wdata = 1 代表 testbench 進入 ap_start 狀態，進入 COMPUTE 準備接收 streaming data，否則維持 NPUT_COEFFICIENT
COMPUTE	輸入 streaming data 儲存到 bram，同時進行 FIR 運算，並在每筆資料運算完成後將 sm_tvalid 拉到 1 讓 testbench 檢查	若沒有算完 600 筆資料的輸出，則維持在 COMPUTE，否則算完後進入 DONE
DONE	將 rdata 設為 2(ap_done)，代表已算好 600 筆 pattern 的輸出	若 rdata 以拉至 2，便回到 IDLE，否則維持 DONE

III. Resource Usage

(1) FF and LUT:

1. Slice Logic

Site Type	Used	Fixed	Prohibited	Available	Util%
Slice LUTs*	258	0	0	53200	0.48
LUT as Logic	258	0	0	53200	0.48
LUT as Memory	0	0	0	17400	0.00
Slice Registers	237	0	0	106400	0.22
Register as Flip Flop	237	0	0	106400	0.22
Register as Latch	0	0	0	106400	0.00
F7 Muxes	0	0	0	26600	0.00
F8 Muxes	0	0	0	13300	0.00

Fig 3 FF and LUT resource usage

(2) Bram

2. Memory

Site Type	Used	Fixed	Prohibited	Available	Util%
Block RAM Tile	0	0	0	140	0.00
RAMB36/FIFO*	0	0	0	140	0.00
RAMB18	0	0	0	280	0.00

Fig 4 BRAM resource usage

(3) DSP

3. DSP

Site Type	Used	Fixed	Prohibited	Available	Util%
DSPs	3	0	0	220	1.36
DSP48E1 only	3				

Fig 5 DSP resource usage

IV. Timing Report

Clock Period: 7.2ns

Critical Path:

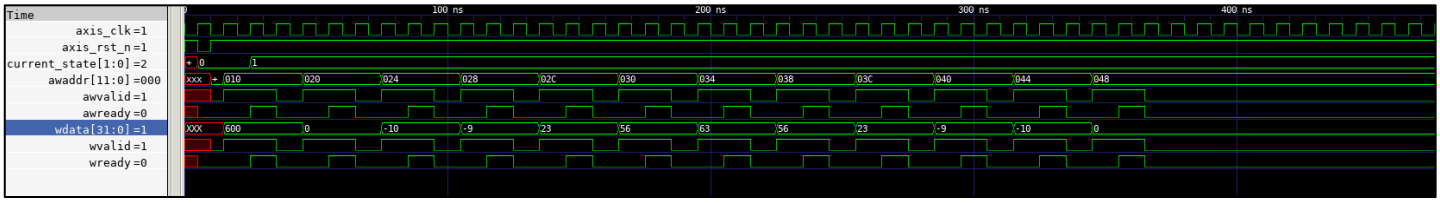
```
Max Delay Paths
--------------------------------------------------------------------------------------
Slack (MET) :             0.076ns  (required time - arrival time)
  Source:                 streaming_data_reg[16]/C
                            (rising edge-triggered cell FDCE clocked by axis_clk  {rise@0.000ns fall@3.600ns period=7.200ns})
  Destination:            single_multi_element0__1/PCIN[0]
                            (rising edge-triggered cell DSP48E1 clocked by axis_clk  {rise@0.000ns fall@3.600ns period=7.200ns})
  Path Group:             axis_clk
  Path Type:              Setup (Max at Slow Process Corner)
  Requirement:            7.200ns  (axis_clk rise@7.200ns - axis_clk rise@0.000ns)
  Data Path Delay:        5.544ns  (logic 4.689ns (84.582%)  route 0.855ns (15.418%))
  Logic Levels:           1  (DSP48E1=1)
  Clock Path Skew:        -0.145ns (DCD - SCD + CPR)
    Destination Clock Delay (DCD):    2.128ns = ( 9.328 - 7.200 )
    Source Clock Delay      (SCD):    2.456ns
    Clock Pessimism Removal (CPR):    0.184ns
  Clock Uncertainty:      0.035ns  ((TSJ^2 + TIJ^2)^1/2 + DJ) / 2 + PE
    Total System Jitter     (TSJ):    0.071ns
    Total Input Jitter      (TIJ):    0.000ns
    Discrete Jitter          (DJ):    0.000ns
    Phase Error              (PE):    0.000ns
```

Fig 6 Max Delay Paths

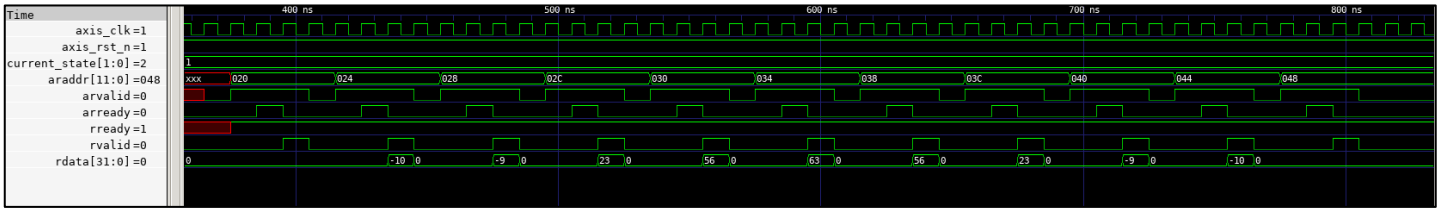
V. Simulation Waveform

1. AXI-Lite

(1) Coefficient Program

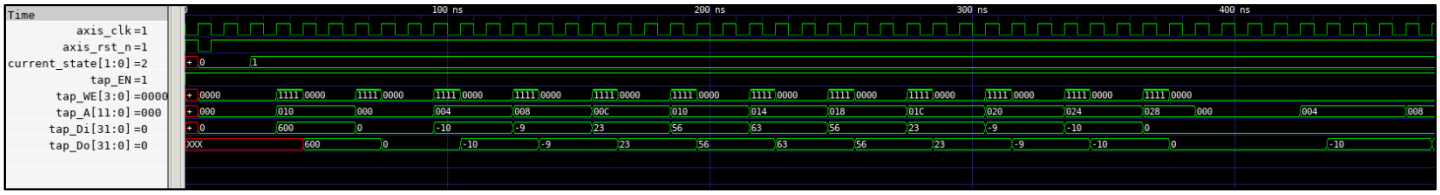


(2) Coefficient Read back

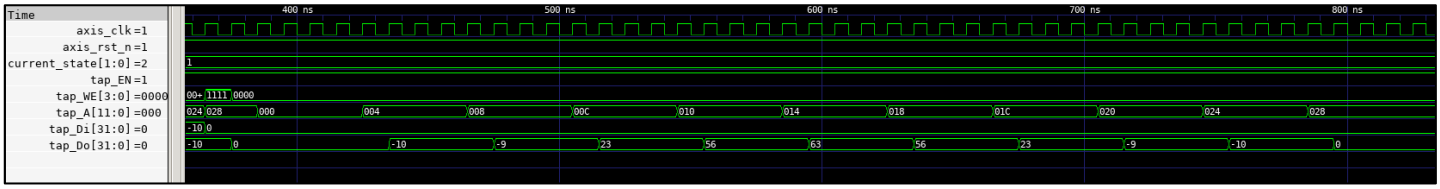


2. RAM Access Control (Tap RAM)

(1) Write

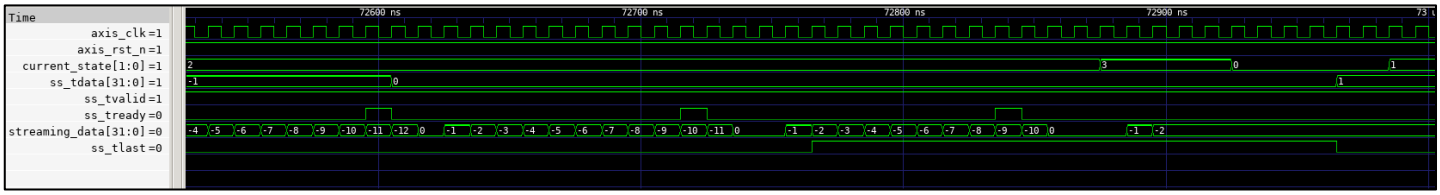


(2) Read

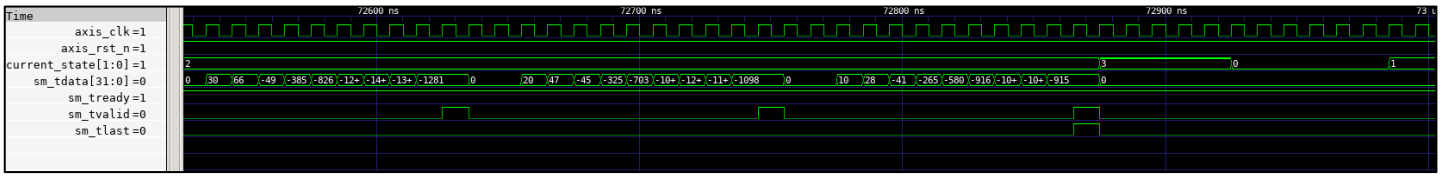


3. AXI-Streaming

(1) Data-in stream-in

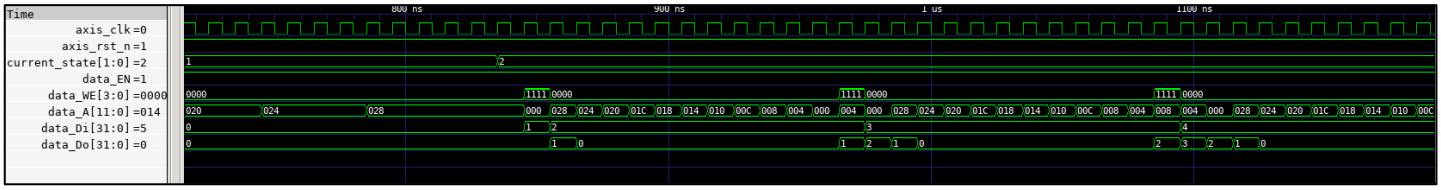


(2) Data-out stream-out

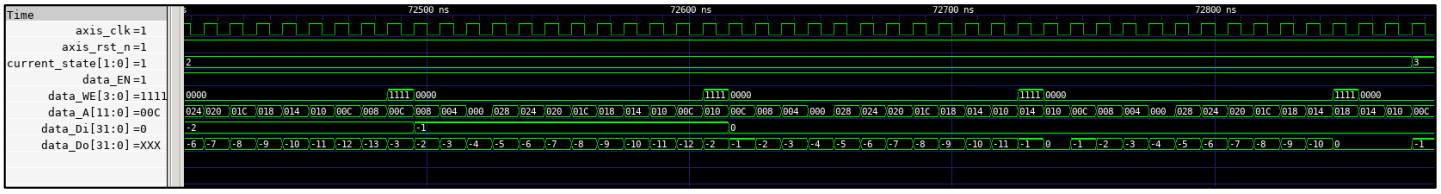


4. RAM Access Control (Data RAM)

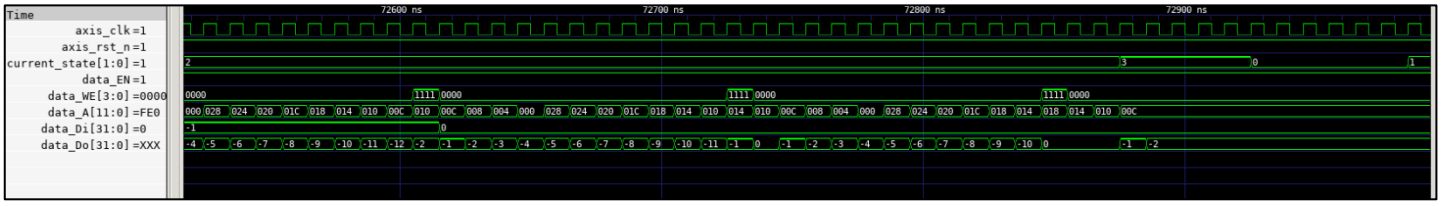
(1) First data input



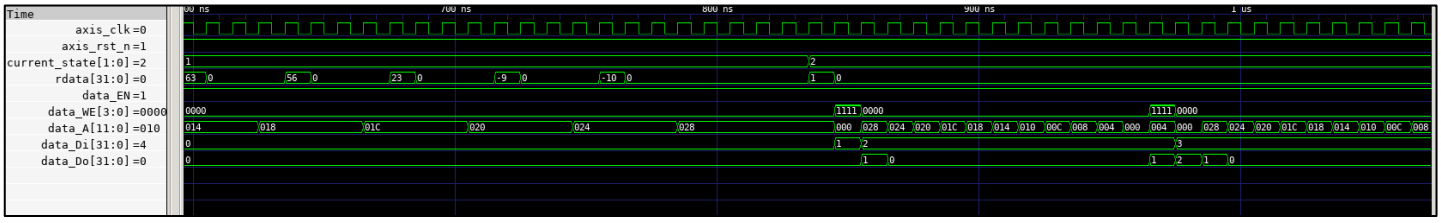
(2) Last Data Input



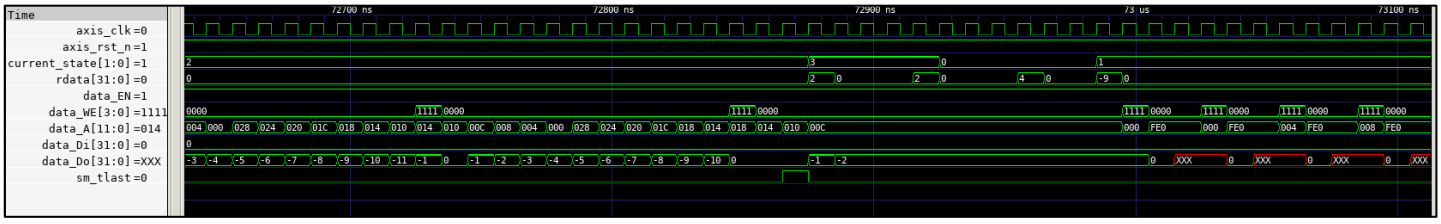
(3) Last Data Output



5. FSM



state 切換成 2，進入 compute 狀態，同時 rdata 拉至 1，代表 ap_start＝1，隨降回 0。



sm_tlast 拉至 1 代表算完 600 筆資料，rdata 拉至 2，代表 ap_done。