

SOC Design

Lab-3 FIR

110590022 陳冠晰

● Function specification

$$\bullet y[t] = \sum (h[i] * x[t - i])$$

In signal processing, a finite-impulse-response, or finite impulse response models are generally linear dynamic models characterized by finite-order moving average representations, implying that their responses to impulse inputs go to zero after a finite number of time steps, equal to the model's memory length.

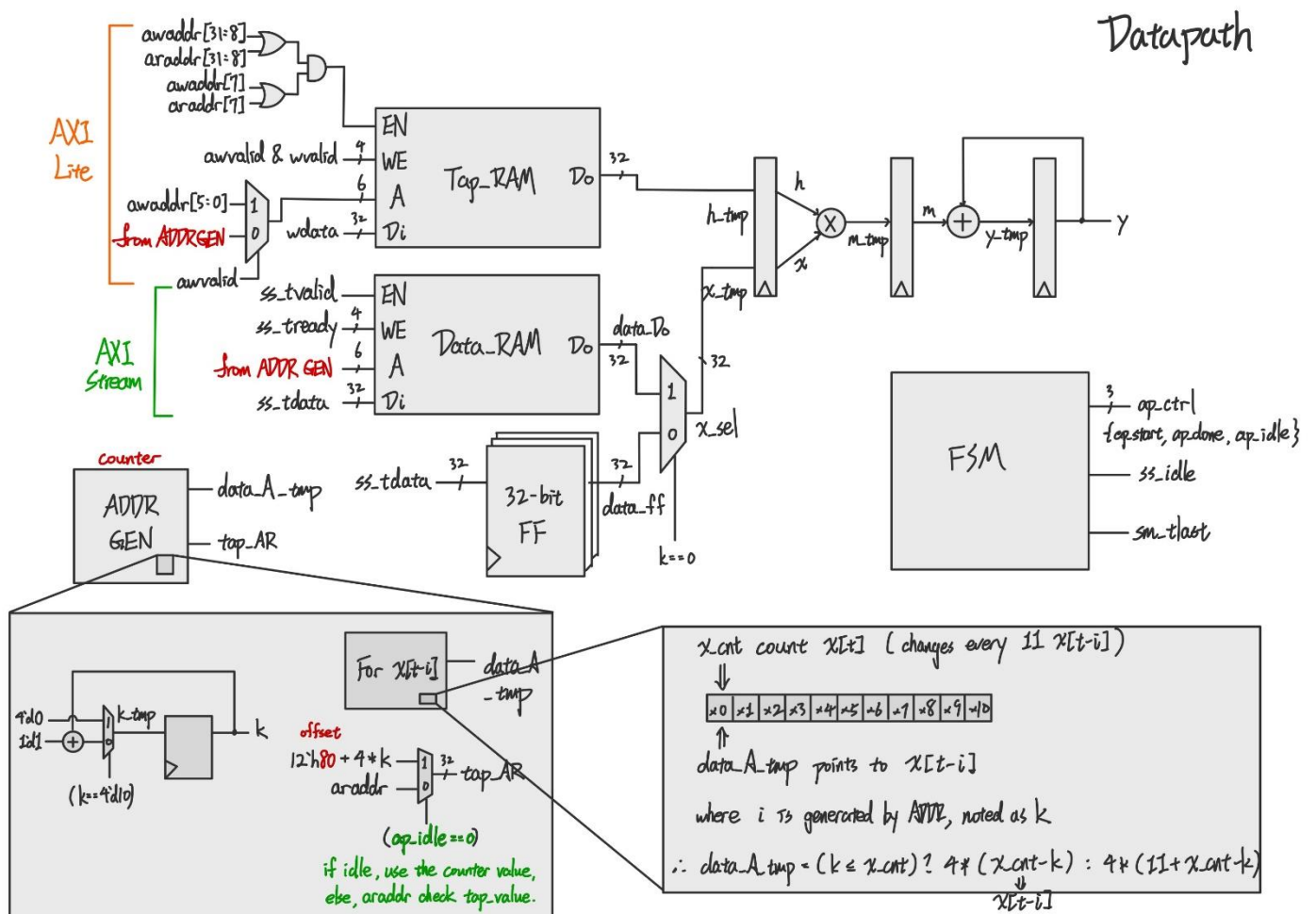
FIR filters can be discrete-time or continuous-time, and digital or analog.

$$\begin{aligned} y[n] &= b_0 x[n] + b_1 x[n - 1] + \cdots + b_N x[n - N] \\ &= \sum_{i=0}^N b_i \cdot x[n - i], \end{aligned}$$

Where:

- . $x[n]$ is the input signal,
- . $y[n]$ is the output signal,
- . N is the filter order; an N th-order filter has $N + 1$ terms on the right-hand side
- . b_i is the value of the impulse response at the i th instant for $0 \leq i \leq N$ of an N th-order FIR filter. If the filter is a direct form FIR filter, then b_i is also a coefficient of the filter.

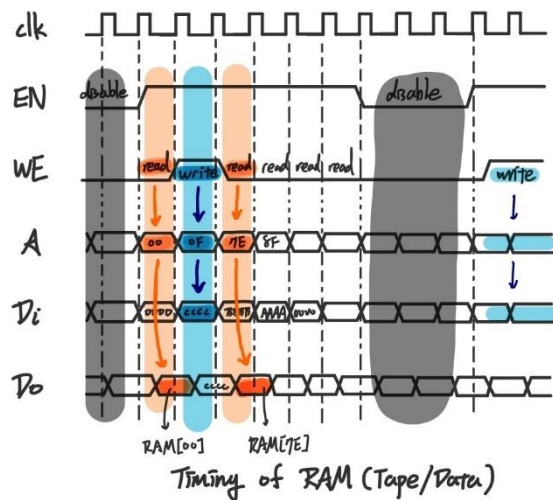
● Block Diagram



Tap_RAM / Data_RAM



32-bit \leftrightarrow 4 byte
 \downarrow
 reg [3:0] RAM[depth] \Rightarrow RAM[A>>2] $(\div 4)$
 A[10:0] \rightarrow in order to access byte



TapeRAM:

8'b10000000

Tape address: 0x80 - 0xFF

$$EN = (AW[3:8] == 0) \mid (AR[3:8] == 0) \\ \& (AW[7] \mid AR[7])$$

Check if read/write address belongs to TapeRAM

$$WE = (AW_{valid} \& \& W_{valid} == 1) ? 4'b1111 : 4'b0000;$$

Check if address/data write is valid

$$A = AW[5:0] \quad AW \text{ will } >> 2, 4\text{-bit left, still can represent the address of 11 tapes}$$

$$Di = Wdata \quad \text{Data-in equals to wdata of AXI-Lite (hits) flow through TapeRAM by axilite}$$

$$Ar_{ready} = 1 \quad (\text{RAM 尚有空间})$$

$$wr_{ready} = 1 \quad (\text{Data Buffer 尚有 space})$$

DataRAM:

$$EN = ss_tvalid$$

$$WE = (ss_tready \& ss_idle) ? 4'b1111 : 4'b0000; \quad (ss_tlast = 0) \quad (\text{Ready to write "New value" \& not finish})$$

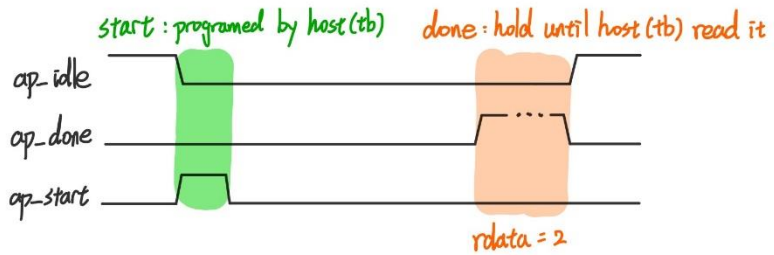
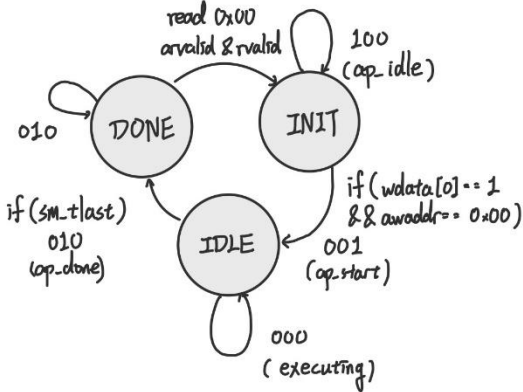
$$A = (ap_ctrl[2] \& \& init_addr < 6d44) ? init_addr : data_A_tmp;$$

If ap_idle == 1, initialize the value in dataRAM.

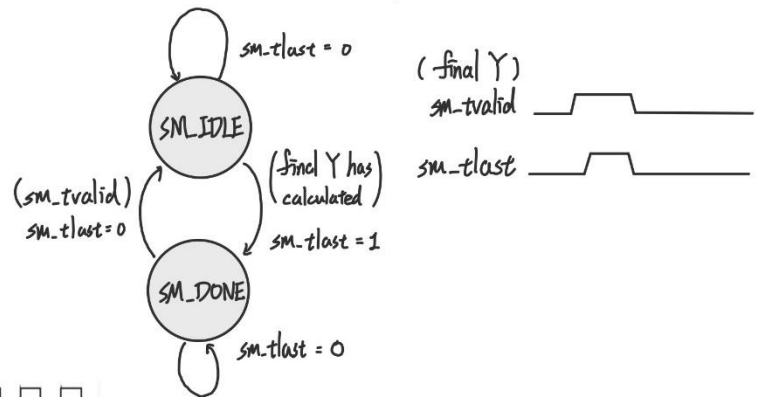
$$Di = ss_tdata$$

FSM

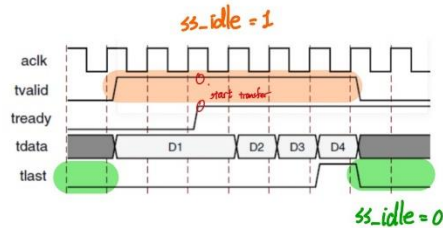
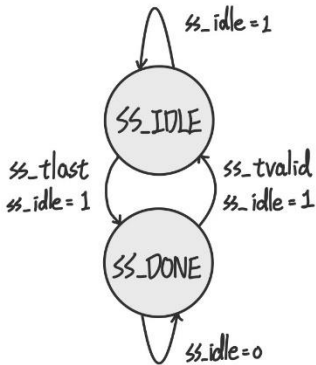
FSM for ap_ctrl:
(ap_start, ap_done, ap_idle)



FSM for sm_tlast control (Last Y calculated)



FSM for ss_idle (Data stream-in),
control the write enable of data_RAM



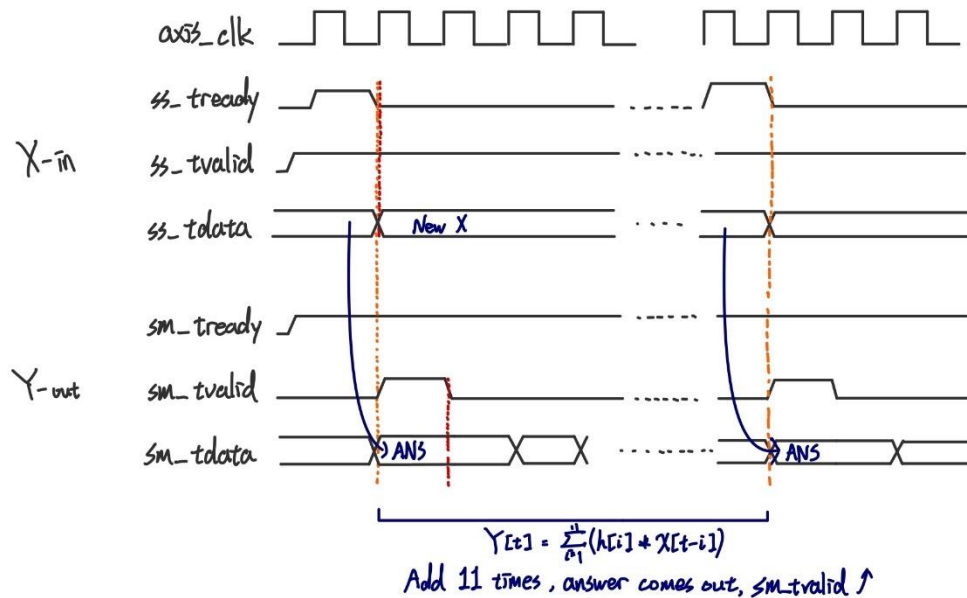
1. 左上角的 FSM 是用來給 ap_start, ap_done, ap_idle 的，一開始處在 INIT state，ap_ctrl = {ap_idle, ap_done, ap_start} = 3'b100，當我們 Host 端 (testbench) program ap_start，代表 FIR 開始運作，ap_idle 降下，進到 IDLE state，ap_ctrl = {ap_idle, ap_done, ap_start} = 3'b000。當我們完成最後一個 Y 的計算，傳出去給 testbench 比對，同時拉高 sm_tlast，就代表 FIR 做完運算，進到 DONE state，ap_ctrl = {ap_idle, ap_done, ap_start} = 3'b010。由於 testbench 需要讀取 ap_done 的訊號，所以要等到 read address == 0x00 讀取到 ap_done 過後再回到 initial 的 state。
2. 左下角的 FSM 專門產生 ss_idle 的訊號拿去 data_RAM 當成 write enable，在還未收到 ss_tlast 的期間，都會是 1，等到收到 ss_tlast，才會進到 SS_DONE state，ss_idle = 0，代表不能再寫了。
3. 右邊的 FSM 是在產生 sm_tlast，當最後一個 Y 計算完，counter 數到 data length，就進到 SM_DONE state，sm_tlast 拉高一個 cycle。

● Operation explaining

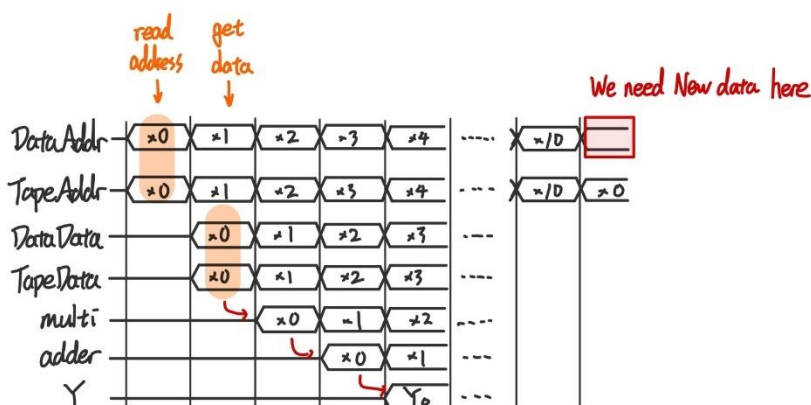
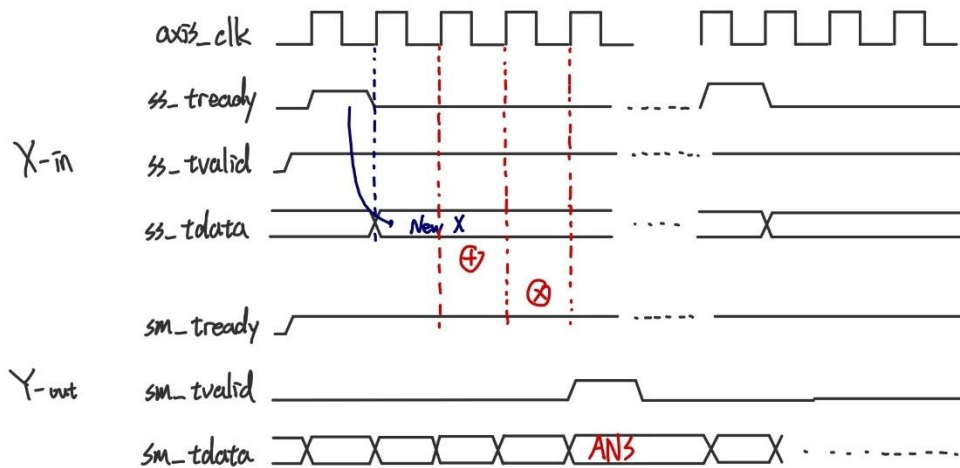
圖片中上半部分是還未做 operation pipeline 會呈現出的 y 波形圖。

ss_tready 每次拉高，代表 stream-in 新的 x[t]進來 RAM。再下個 cycle 可以發現 sm_tvalid 拉高，代表計算出的 Y 被拿去 testbench 做比對。

AXI-Stream timing waveform (Before Operation Pipeline)



AXI-Stream timing waveform (After Operation Pipeline)



上圖最下面是做完 operation pipeline 後呈現出的 timing waveform，可以看到中間圖片中，X stream-in 和 Y stream-out 間變化。

- **Resource usage**

Site Type	Used	Fixed	Prohibited	Available	Util%
Slice LUTs*	190	0	0	53200	0.36
LUT as Logic	190	0	0	53200	0.36
LUT as Memory	0	0	0	17400	0.00
Slice Registers	224	0	0	106400	0.21
Register as Flip Flop	224	0	0	106400	0.21
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

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

- **Timing report**

Setup	Hold	Pulse Width
Worst Negative Slack (WNS): 0.067 ns	Worst Hold Slack (WHS): 0.140 ns	Worst Pulse Width Slack (WPWS): 4.500 ns
Total Negative Slack (TNS): 0.000 ns	Total Hold Slack (THS): 0.000 ns	Total Pulse Width Negative Slack (TPWS): 0.000 ns
Number of Failing Endpoints: 0	Number of Failing Endpoints: 0	Number of Failing Endpoints: 0
Total Number of Endpoints: 540	Total Number of Endpoints: 540	Total Number of Endpoints: 225

All user specified timing constraints are met.

Timing												
Intra-Clock Paths - fir_timing - Setup												
Name	Slack	Levels	Routes	High Fanout	From	To	Total Delay	Logic Delay	Net Delay	Requirement	Source Clock	Destination Clock
Path 1	0.067	9	10	72	y_cnt_reg[2]/C	rdata[1]	8.441	4.838	3.603	10.0	fir_timing	fir_timing
Path 2	0.075	9	10	72	y_cnt_reg[2]/C	sm_tlast	8.433	4.830	3.603	10.0	fir_timing	fir_timing
Path 3	0.797	7	8	1	h_reg[16]/C	m_reg[29]/D	9.098	7.435	1.663	10.0	fir_timing	fir_timing
Path 4	0.803	7	8	1	h_reg[16]/C	m_reg[31]/D	9.092	7.429	1.663	10.0	fir_timing	fir_timing
Path 5	0.878	7	8	1	h_reg[16]/C	m_reg[30]/D	9.017	7.354	1.663	10.0	fir_timing	fir_timing
Path 6	0.902	7	8	1	h_reg[16]/C	m_reg[28]/D	8.993	7.330	1.663	10.0	fir_timing	fir_timing
Path 7	0.914	6	7	1	h_reg[16]/C	m_reg[25]/D	8.981	7.318	1.663	10.0	fir_timing	fir_timing
Path 8	0.920	6	7	1	h_reg[16]/C	m_reg[27]/D	8.975	7.312	1.663	10.0	fir_timing	fir_timing
Path 9	0.966	7	8	84	data_length_reg[3]/C	data_A[2]	7.542	4.743	2.799	10.0	fir_timing	fir_timing
Path 10	0.966	7	8	84	data_length_reg[3]/C	data_A[3]	7.542	4.743	2.799	10.0	fir_timing	fir_timing

