

# EESM 6000C System on chip laboratory

## Lab-3 FIR Report

21106974 SHEN Zijun

### ● 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.

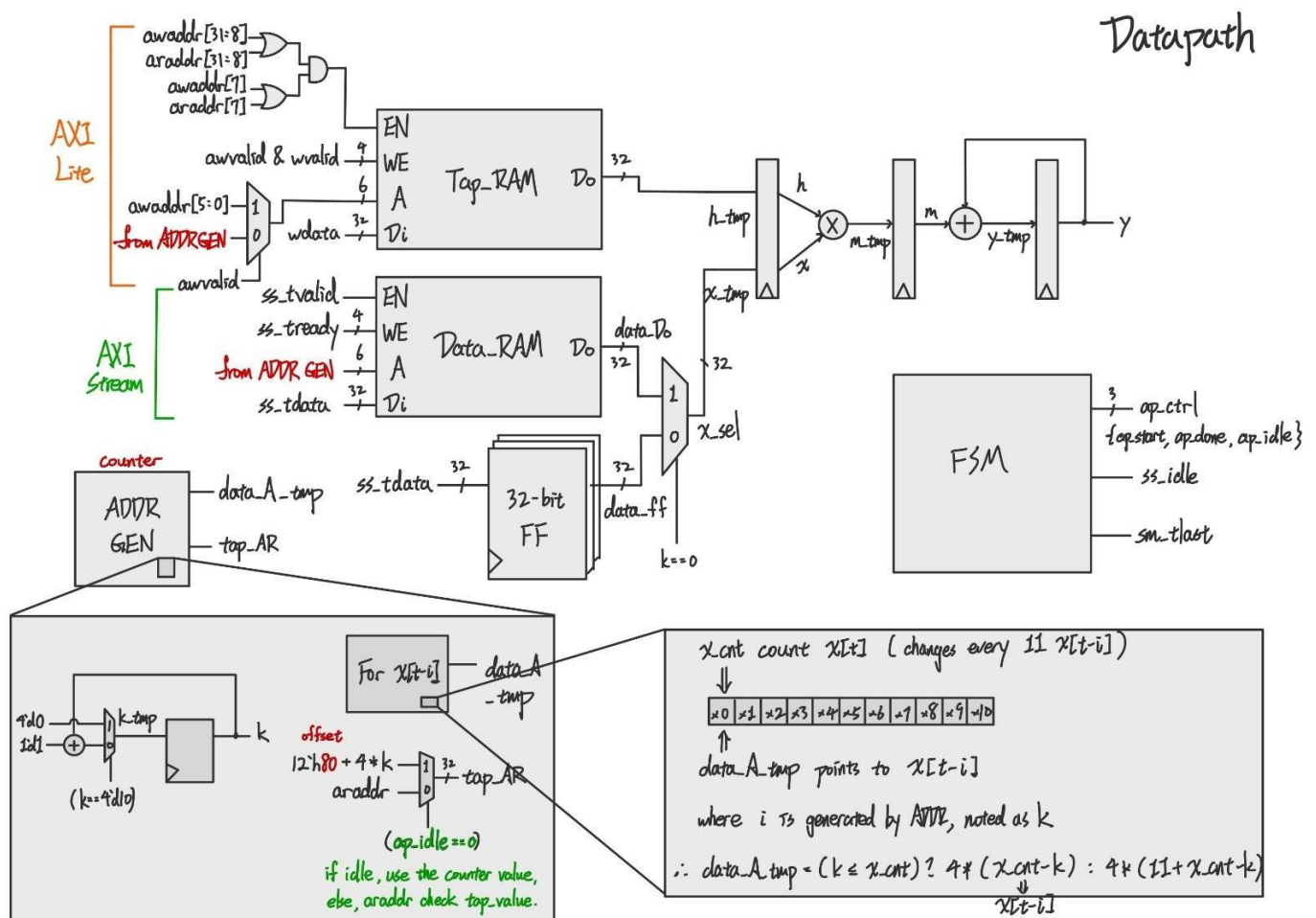
FIR filters can be discrete-time or continuous-time, and digital or analog.

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

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.

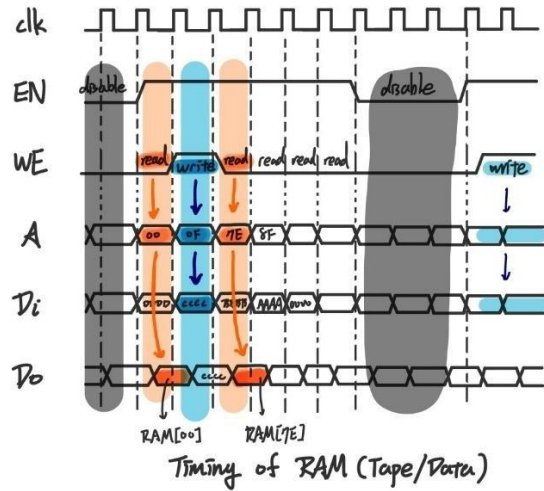
## ● Block Diagram



# 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'b/00000000  
 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

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

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

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

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

## DataRAM:

$$EN = ss\_tvalid$$

$$WE = (ss\_tready \& ss\_idle) ? 4'b1111 : 4'b0000; \quad (ss\_tlast = 0) \quad (\text{Ready to write New value \& not finish})$$

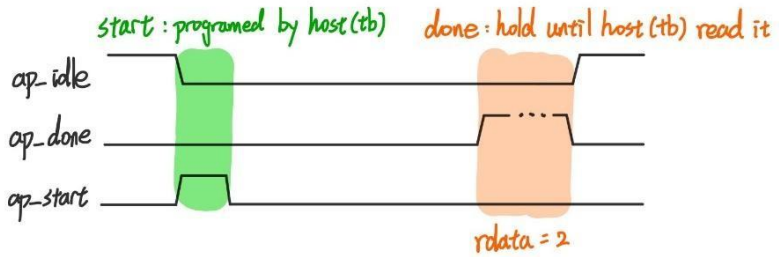
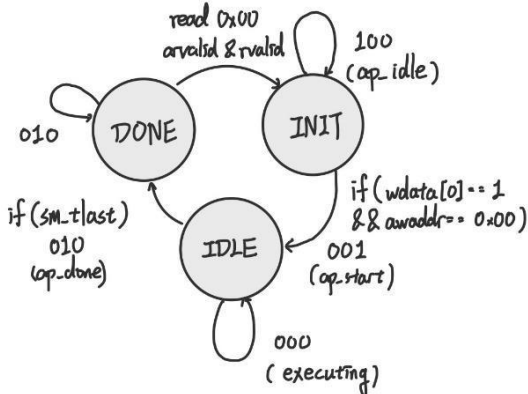
$$A = (ap\_ctrl[2] \& \& init\_addr < 6d44) ? init\_addr : data\_A\_tmp;$$

If ap\\_idle == 1, initialize the value in dataRAM.

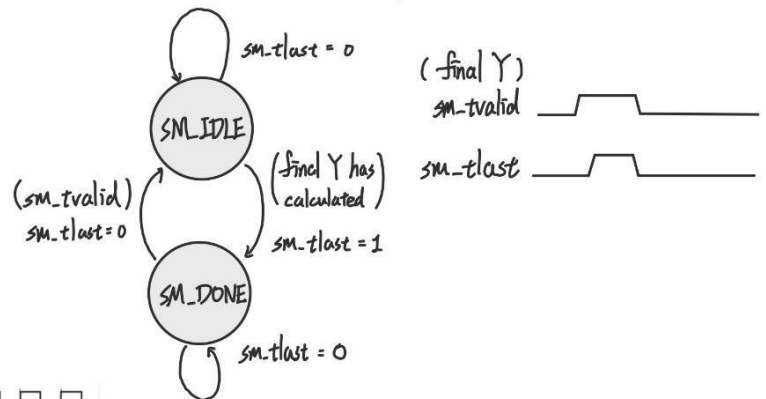
$$Di = ss\_tdata$$

# FSM

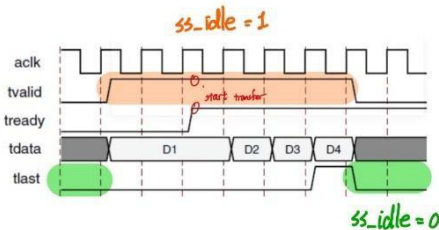
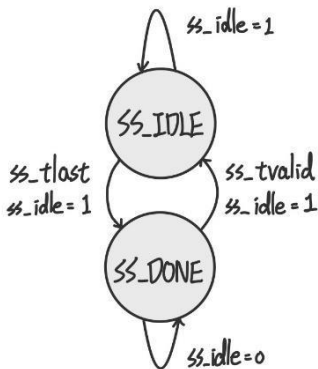
FSM for ap\_ctrl:  
(ap\_start, ap\_done, ap\_idle)



FSM for sm\_tlast control (Last Y calculated)



FSM for ss\_idle (Data stream-in),  
control the write enable of data\_RAM

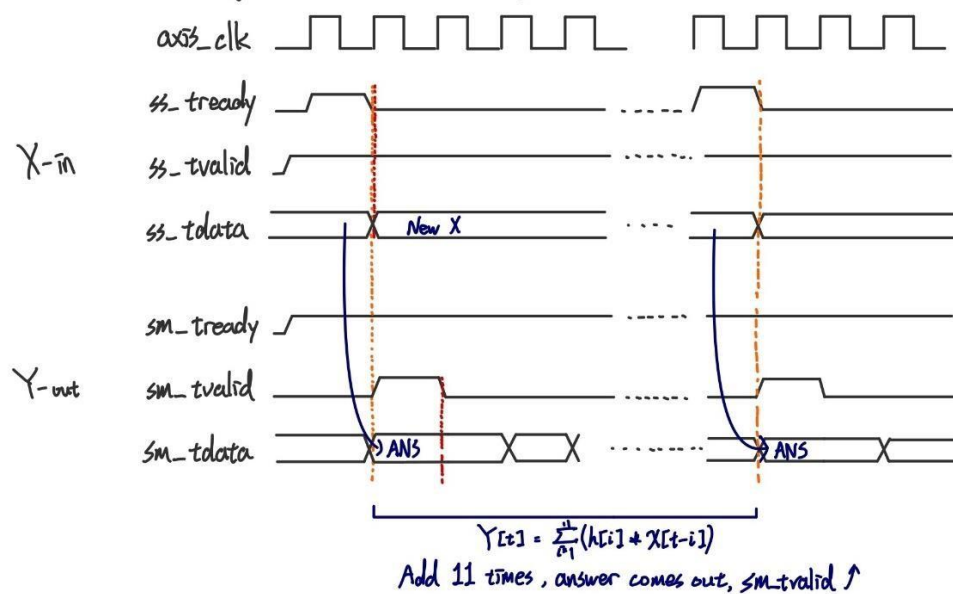


1. The FSM in the top left is given to ap\_start, ap \_ stone, ap \_ dle, starting at INIT state, ap\_ctrl = {ap\_dle, ap\_stone, ap\_start} = 3'b100, when we Host end (testbench) program ap\_start, representing FIR, ap\_idle, down to IDLE state, ap\_ctrl = {ap\_idle, ap\_done, ap\_start} = 3'b000. When we finish the calculation of the last Y, send it to testbench alignment, and raise sm\_tlast, FIR completes the calculation and goes to DONE state, ap\_ctrl = {ap\_idle, ap\_done, ap\_start} = 3'b010. Since testbench needs to read the ap\_done signal, wait until read address == 0x00 reads to ap\_done before returning to the initial state.
2. The FSM in the lower left corner specially produces the signal of ss\_iddle to take data \_ RAM as write enable, before the ss\_tlast, it will be 1, until the ss\_tlast, will enter the SS\_DONE state, ss\_idle = 0, means can not write again.
3. The FSM on the right is producing sm\_tlast. When the last Y is calculated, counter reaches data length, it goes to SM\_DONE state, and sm\_tlast raises a cycle.

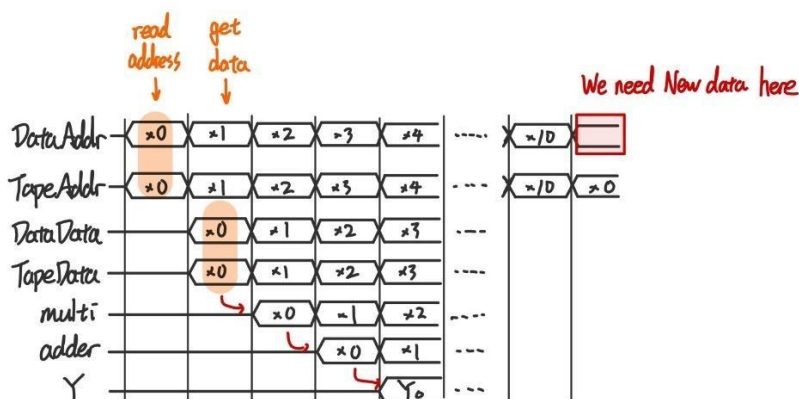
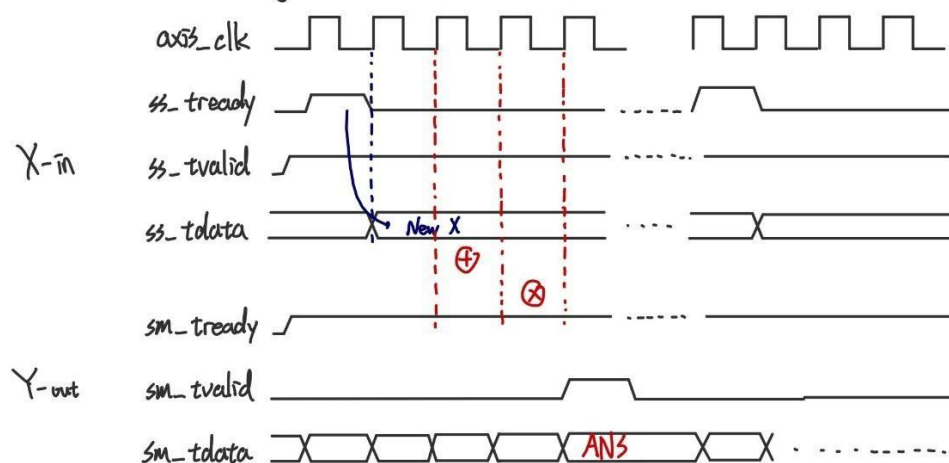
## ● Operation explaining

The upper part of the picture is the y-wave graph that will appear before the operation pipeline. Each time `ss_tready` is pulled high, representing stream-in new  $x[t]$  coming in the RAM. The next cycle can be found `sm_tvalid`, pulled high, represents that the calculated  $Y$  is taken to the testbench for comparison.

AXI-Stream timing waveform (Before Operation Pipeline)



AXI-Stream timing waveform (After Operation Pipeline)



● 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

Design Timing Summary			
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.			

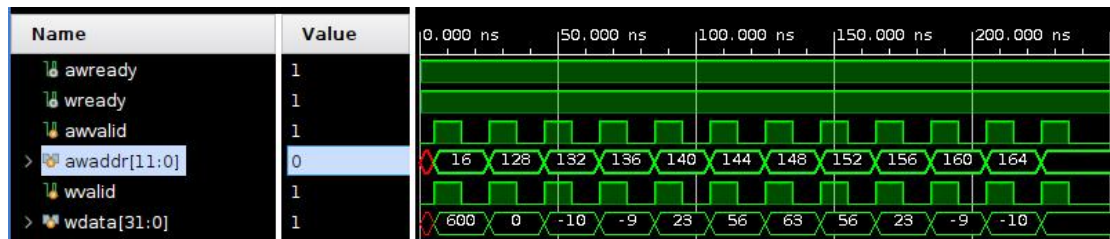
Intra-Clock Paths - fir_timing - Setup												
Name	Slack	Levels	Routes	High Fanout	From	To	Total Delay	Logic Delay	Net Delay	Requirement	Source Clock	Destination Clock
Path 1	0.067	9	10	72	y_cnt_reg[2]/C	rdata[1]	8.441	4.838	3.603	10.0	fir_timing	fir_timing
Path 2	0.075	9	10	72	y_cnt_reg[2]/C	sm_tlast	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



Tcl Console Messages Log Reports Design Runs Timing x													
Intra-Clock Paths - fir_timing - Hold													
Name	Slack	Levels	Routes	High Fanout	From	To	Total Delay	Logic Delay	Net Delay	Requirement	Source Clock	Destination Clock	
Path 11	0.140	1	2	2	ss_state_reg[C]	ss_state_reg[D]	0.384	0.248	0.136	0.0	fir_timing	fir_timing	
Path 12	0.146	1	2	6	init_addr_reg[3]/C	init_addr_reg[3]/D	0.390	0.245	0.145	0.0	fir_timing	fir_timing	
Path 13	0.146	1	2	6	init_addr_reg[4]/C	init_addr_reg[5]/D	0.390	0.245	0.145	0.0	fir_timing	fir_timing	
Path 14	0.147	1	2	7	y_cnt_reg[2]/C	tlast_cnt_reg[0]/D	0.391	0.245	0.146	0.0	fir_timing	fir_timing	
Path 15	0.148	1	2	8	init_addr_reg[2]/C	init_addr_reg[2]/D	0.392	0.245	0.147	0.0	fir_timing	fir_timing	
Path 16	0.151	1	2	8	init_addr_reg[2]/C	init_addr_reg[4]/D	0.395	0.248	0.147	0.0	fir_timing	fir_timing	
Path 17	0.211	1	2	5	tlast_cnt_reg[3]/C	tlast_cnt_reg[4]/D	0.455	0.245	0.210	0.0	fir_timing	fir_timing	
Path 18	0.211	1	2	5	tlast_cnt_reg[8]/C	tlast_cnt_reg[9]/D	0.455	0.245	0.210	0.0	fir_timing	fir_timing	
Path 19	0.213	1	2	6	y_cnt_reg[3]/C	y_cnt_reg[3]/D	0.457	0.245	0.212	0.0	fir_timing	fir_timing	
Path 20	0.213	1	2	6	y_cnt_reg[4]/C	y_cnt_reg[4]/D	0.457	0.245	0.212	0.0	fir_timing	fir_timing	

## ● Simulation waveforms

### ➤ Coefficient



### ➤ X-in/Y-out

