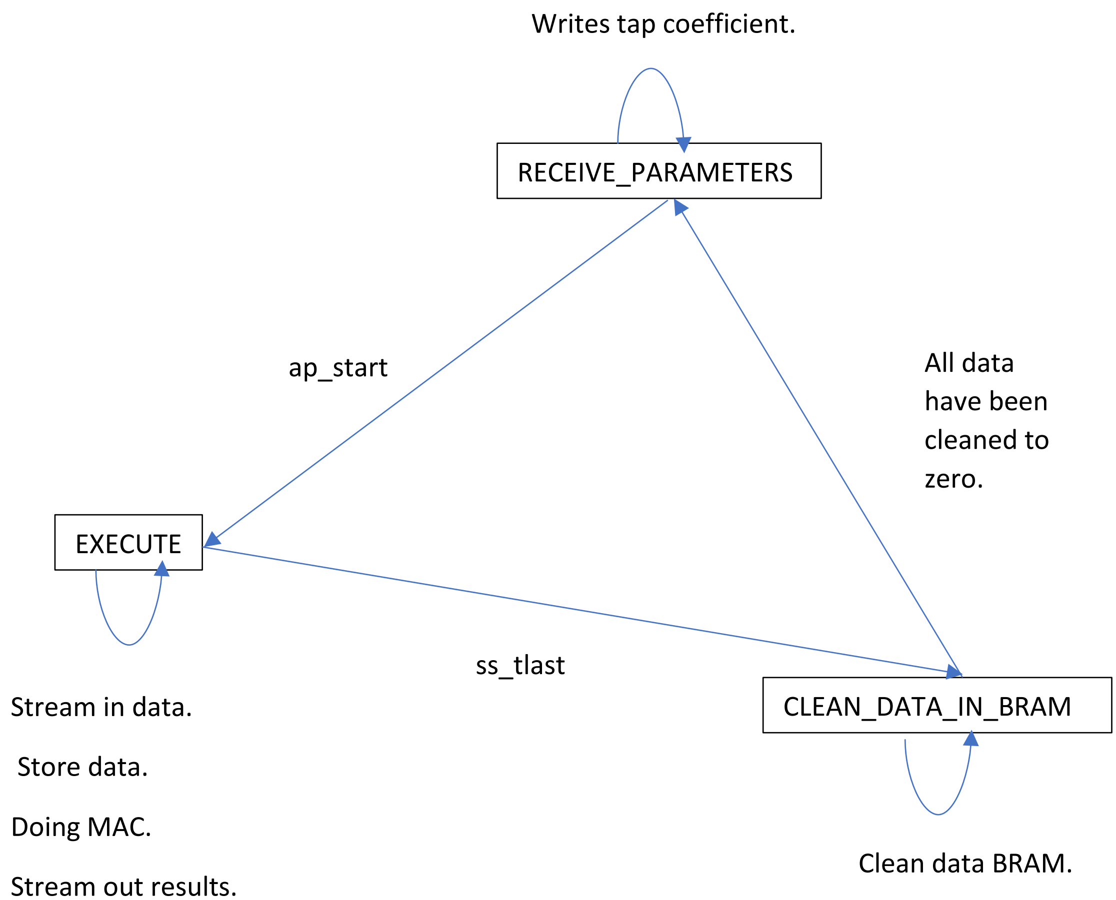
SoC Design Lab 3

R12943013 聶家任

- Block diagram
  - Control and datapath



- Describe operation
  - After reset, the state is in RECEIVE_PARAMETERS. In this state, the circuit receives coefficients such as data length and tap coefficients. My method is zeroing the data BRAM when receiving each tap coefficients. So, when writing the coefficient at 0x20, the circuit clears the data at 0x20. In reality, the receiving address is reduced by ‘h20 to correspond to the address in BRAM.
  - After ap_start, the state becomes EXECUTE and the circuit starts streaming in data, does MAC, stores data into data BRAM and streams out the computed results. My method is to remember the first data position in BRAM and take the data in BRAM based on the (first data position+counter) mod 10, noting that we do not need to store the newest data and read the data from BRAM, we can use the newest data directly. So here is (mod 10) not (mod 11). In the meanwhile, when the newest data is used to compute the FIR result, the newest data can be saved to BRAM. Thus, the time saves.
  - When the circuit is in EXECUTE state and the newest data input is flagged with ss_tlast, the next state is CLEAN_DATA_IN_BRAM. This state is crucial because when new stream starts to get in, we should have the data BRAM empty and have no old data in it.
  - When the data BRAM has been clean to zero, the state transitions from CLEAN_DATA_IN_BRAM to RECEIVE_PARAMETERS to await next ap_start signal or the host wants to change the tap coefficients.
  - The ap_done and ap_idle signals are taken care of by the state. When the state is EXECUTE, the ap_done=0 and ap_idle=0, and the when asked about ap_start, it’s always been cleared to 0 because this application has one host only and it will always start when receiving the ap_start signal.
  - When the circuit is outputting results, be it rvalid or sm_tvalid, the circuit will always wait for the host to respond with ready signals.

- Resource usage

```
Detailed RTL Component Info :
+---Adders :
	   2 Input   12 Bit       Adders := 4
	   2 Input    5 Bit       Adders := 1
	   2 Input    4 Bit       Adders := 2
+---Registers :
	               32 Bit    Registers := 3
	                4 Bit    Registers := 2
	                2 Bit    Registers := 2
	                1 Bit    Registers := 4
+---Multipliers :
	              32x32  Multipliers := 1
+---Muxes :
	   2 Input   32 Bit        Muxes := 18
	   3 Input   32 Bit        Muxes := 1
	   4 Input   32 Bit        Muxes := 4
	   4 Input   12 Bit        Muxes := 3
	   2 Input   12 Bit        Muxes := 12
	   2 Input    6 Bit        Muxes := 5
	   4 Input    4 Bit        Muxes := 5
	   2 Input    4 Bit        Muxes := 10
	   4 Input    2 Bit        Muxes := 2
	   2 Input    2 Bit        Muxes := 1
	   5 Input    2 Bit        Muxes := 1
	   2 Input    1 Bit        Muxes := 34
	   4 Input    1 Bit        Muxes := 21
```

- Timing report

Design Timing Summary

Setup		**Hold**		**Pulse Width**	
Worst Negative Slack (WNS):	0.781 ns	Worst Hold Slack (WHS):	-0.006 ns	Worst Pulse Width Slack (WPWS):	2.725 ns
Total Negative Slack (TNS):	0.000 ns	Total Hold Slack (THS):	-0.024 ns	Total Pulse Width Negative Slack (TPWS):	0.000 ns
Number of Failing Endpoints:	0	Number of Failing Endpoints:	6	Number of Failing Endpoints:	0
Total Number of Endpoints:	218	Total Number of Endpoints:	218	Total Number of Endpoints:	125

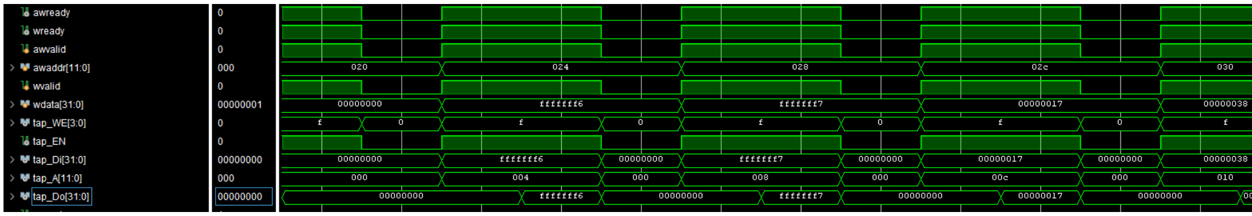
Timing constraints are not met.

Note that the hold time violation is tolerable because it can be handled when doing placement and route. The slack is 0.781 ns when I set the clock cycle to 6 ns.

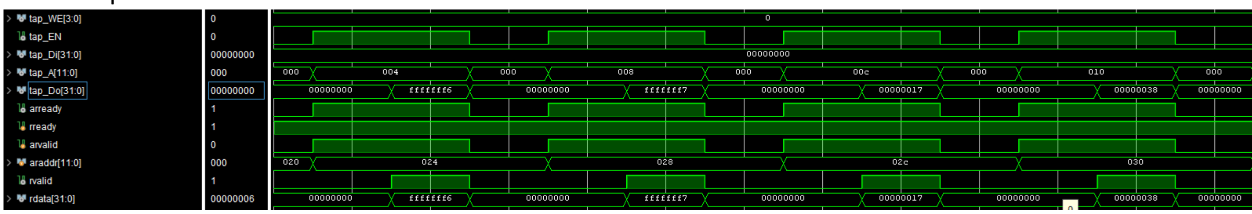
Name	Slack ^1	Levels	Routes	High Fanout	From	To	Total Delay	Logic Delay	Net Delay	Requirement
Path 1	0.781	14	15	51	counter_r_reg[0]/C	fir_result_r_reg[31]/D	5.083	4.077	1.006	6.0

The max delay path is from my counter comparing whether it's the time to use the newest data to the FIR result computed by MAC operation.

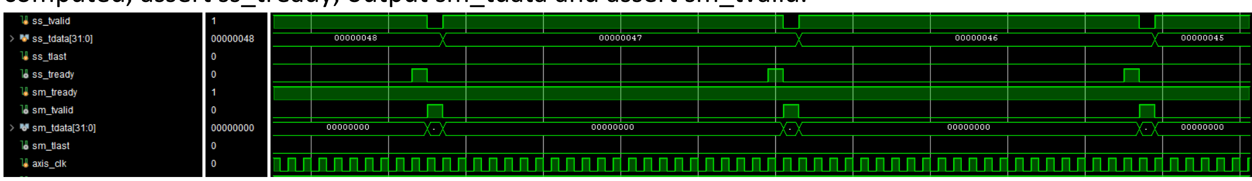
- Waveform
  - Coefficient write(write BRAM): When awaddr and wdata are obtained the circuit suddenly raises write enable signal and writes the tap coefficient into tap BRAM.



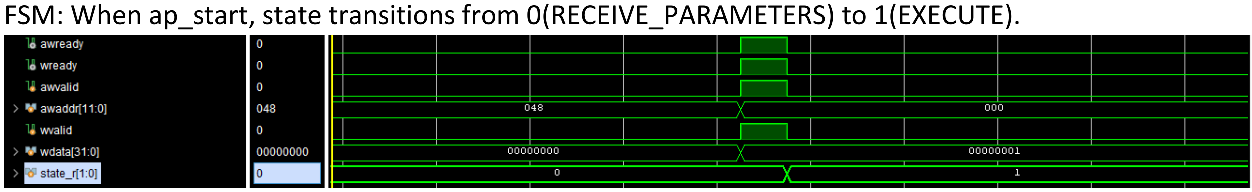
  - Coefficient read back(read BRAM): When arvalid is asserted, the circuit asks the tap coefficient in the corresponding address in BRAM and raise the rvalid signal when the BRAM reponds.



  - Data in & Data out: When ss_tvalid, start to compute the fir result, and after fir result is computed, assert ss_tready, output sm_tdata and assert sm_tvalid.



  - FSM: When ap_start, state transitions from 0(RECEIVE_PARAMETERS) to 1(EXECUTE).



    When ss_tlast is asserted, assert sm_tlast as the fir result is valid and state transition from 1(EXECUTE) to 2(CLEAN_DATA_IN_BRAM).