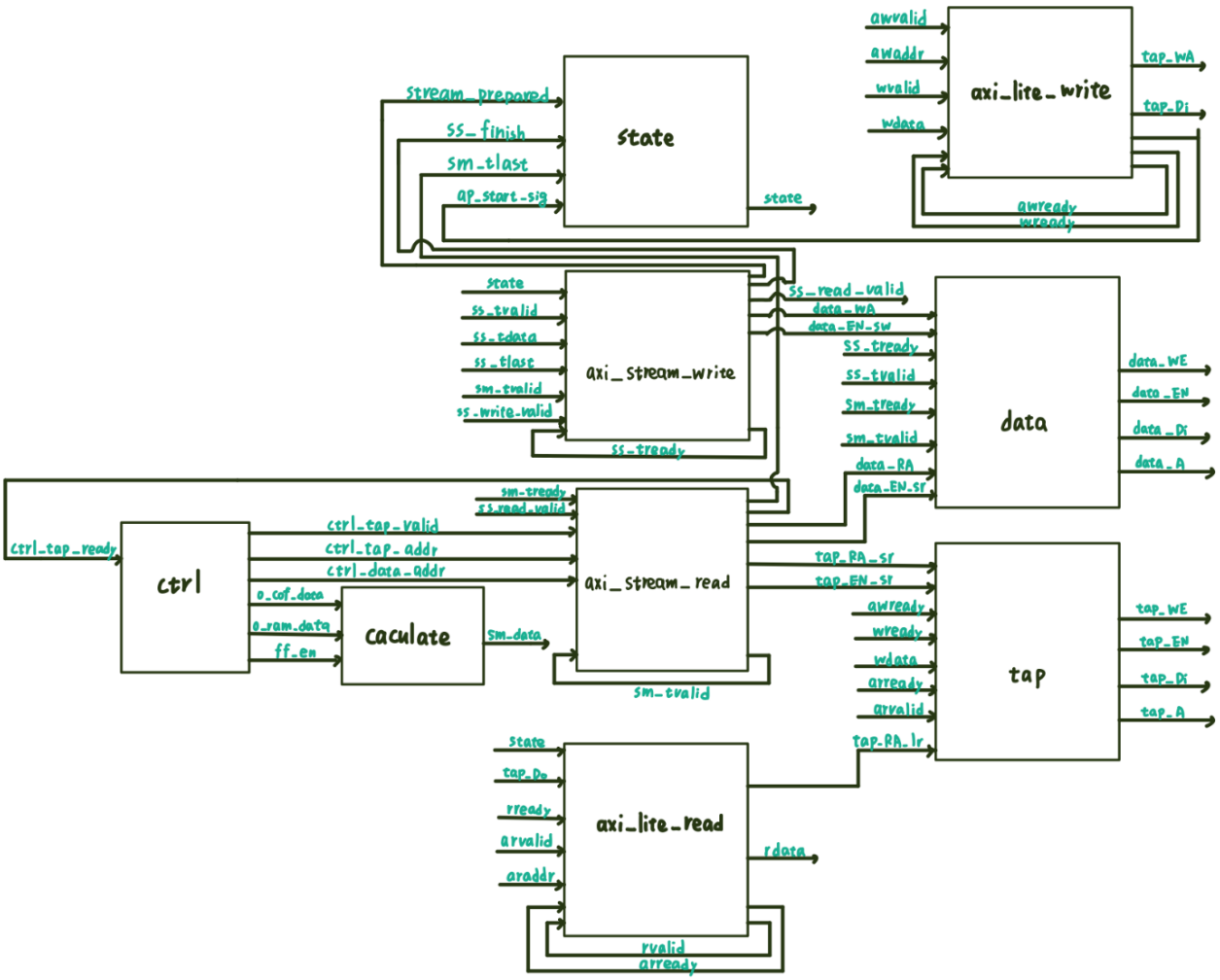
Lab#3 Report

台科大 M11207002 陳泊佑

- **Block diagram**



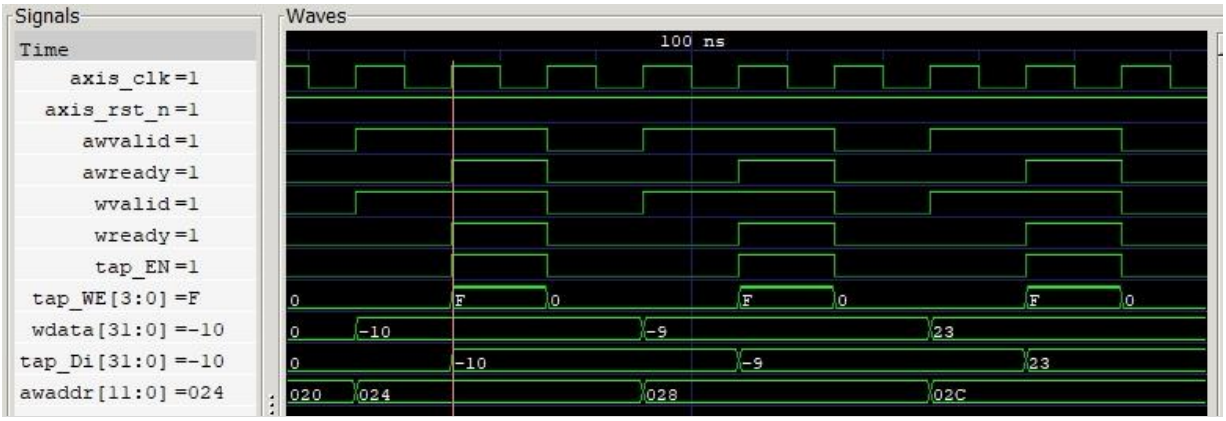
● Describe operation

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

Tap-parameters

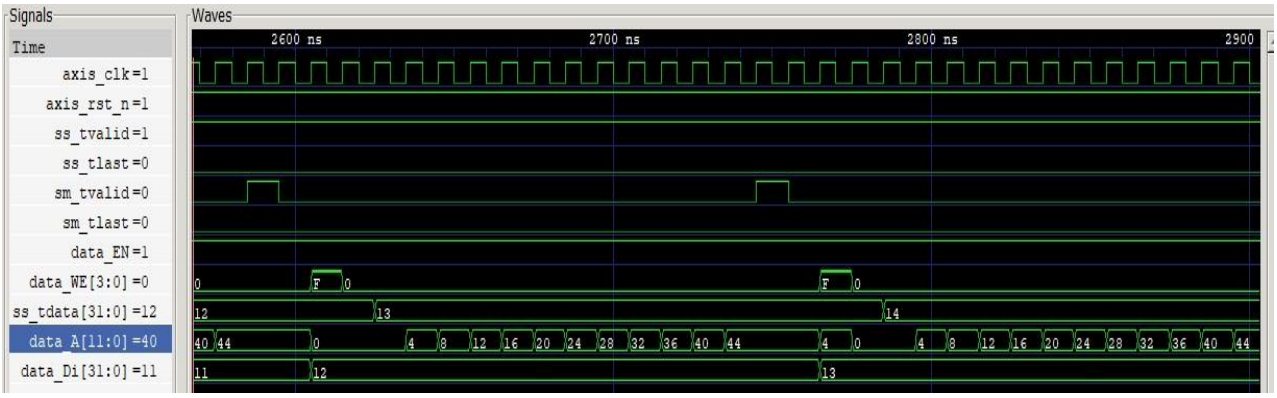
當 awvalid 跟 wvalid 等於 1 時，準備要把 tap 寫入。

當 tap_EN=1 與 tap_WE=4'hF 時，會將 wdata 寫進去 awaddr 的位置。



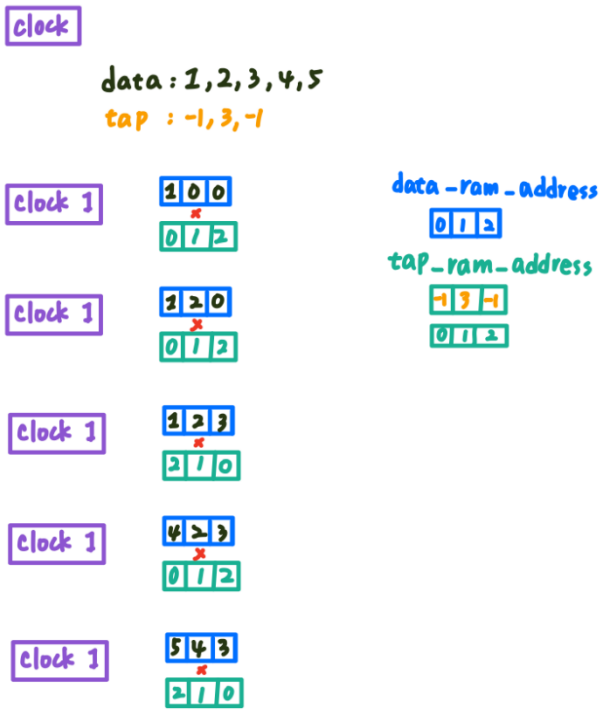
data-in

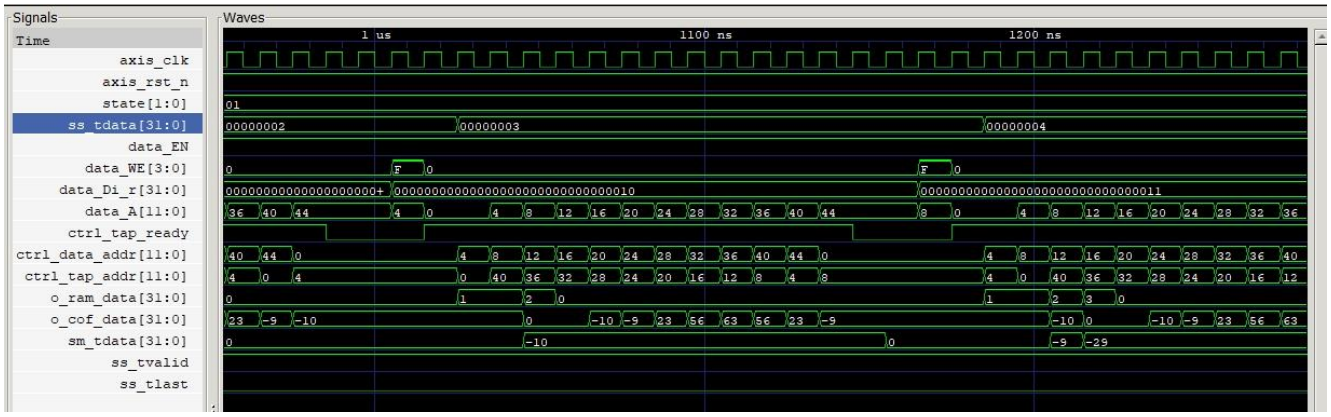
input 透過 stream 傳進來，並先用一個 fifo 來存取 data，要讀資料時，當要讀 data 時，data_EN=1 與 data_WE=4'hF，就會開始將 ss_tdata 寫進去 data_A 的位置。



II. How to access shiftram and tapRAM to do computation

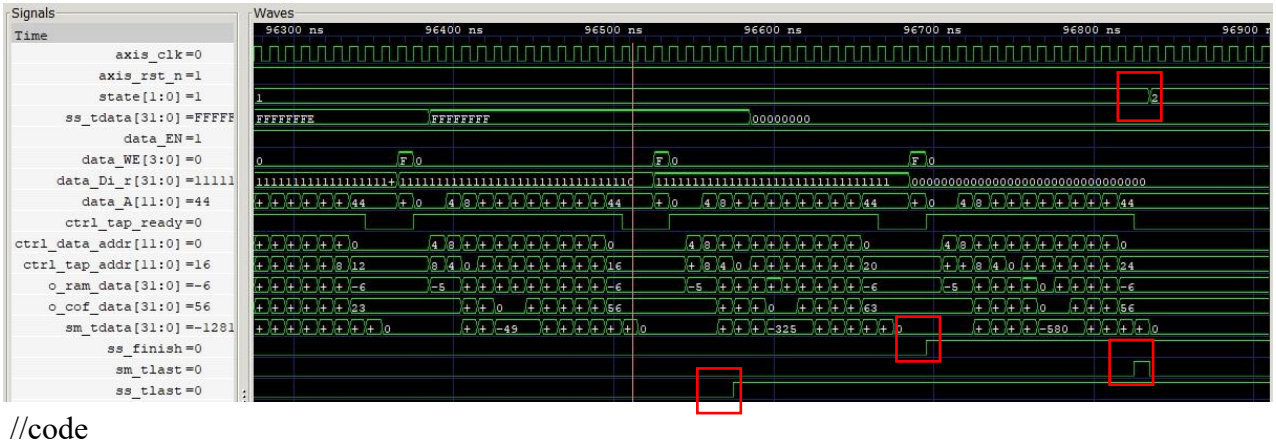
為了不動讀取到的 RAM 位置，這裡選擇控制 tap_addr 跟 data_addr 來讓它們相乘，下面以 data:1,2,3,4,5 以及 tap:-1,3,-1 來做示範





III. How ap_done is generated?

當 ss_tlast 和 sm_tlast 都為 1 後，ap_done=2 跳起來，故這邊會把 state = 2'b10。



//code

```
parameter ap_done = 2;

reg [1:0] state;
wire stream_prepared;
reg ap_start_sig;
reg     sm_tlast_r;
reg     ss_finish_r;
wire    ss_finish;
//state
always @( posedge axis_clk ) begin
    if ( !axis_rst_n ) begin
        state <= ap_idle;
    end
    else begin
        case(state)
            ap_idle: begin
                if(stream_prepared  && ap_start_sig)
                    state <= ap_start;

                sm_tlast_r <= 0;
            end

            ap_start: begin
                if(ss_finish && sm_tlast)
                    state <= ap_done;

                if(ss_finish  && !ctrl_tap_valid)
                    sm_tlast_r <= 1;
                else
                    sm_tlast_r <= 0;

            end
```

- # Resource usage

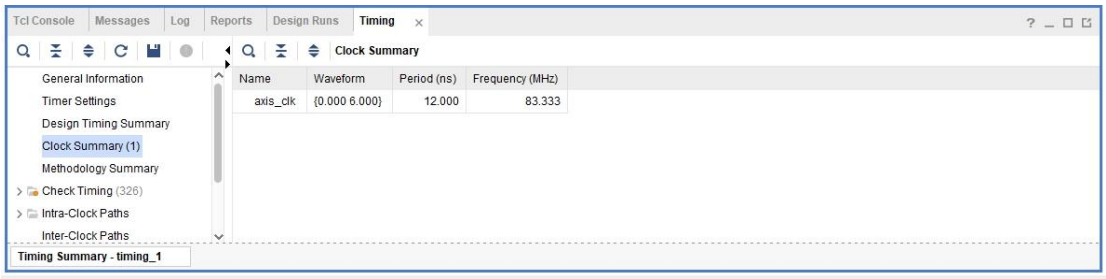
Utilization | **Post-Synthesis** | Post-Implementation

Graph | **Table**

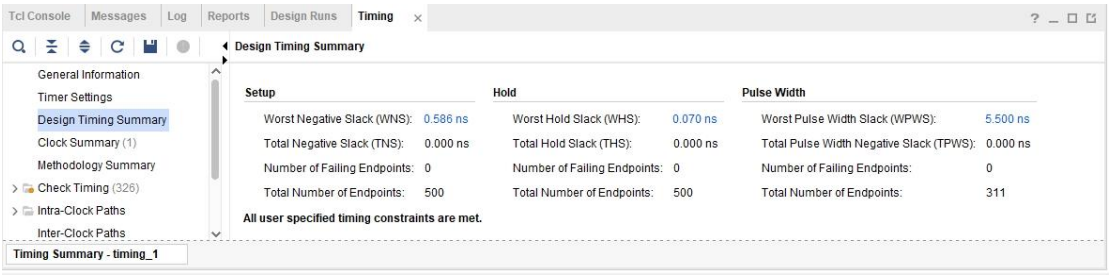
Resource	Estimation	Available	Utilization...
LUT	287	53200	0.54
FF	310	106400	0.29
DSP	3	220	1.36
IO	329	125	263.20
BUFG	1	32	3.13

- # Timing report

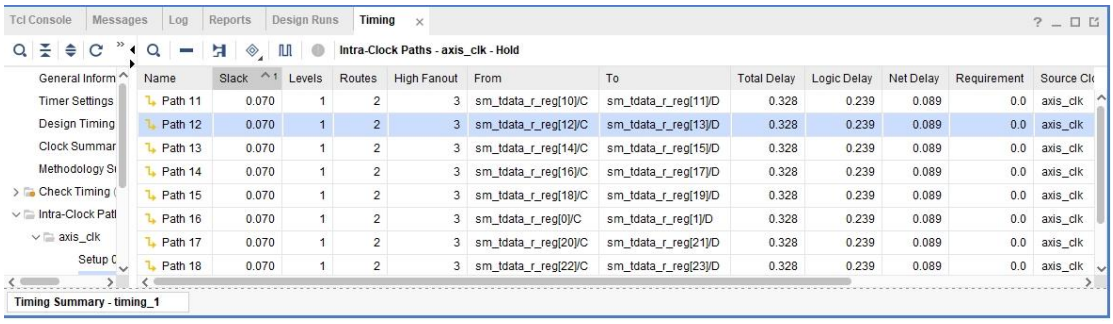
I. Try to synthesize the design with maximum frequency

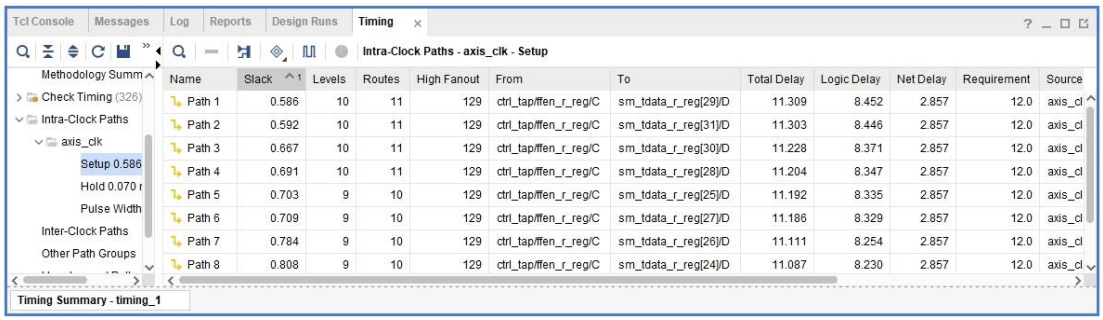


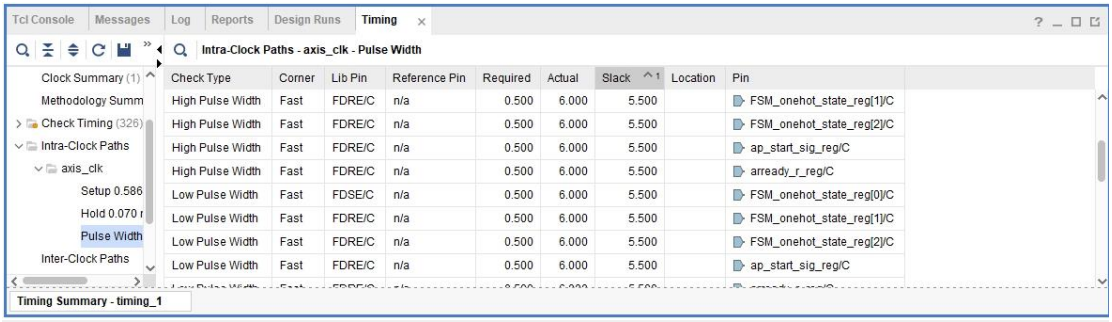
II. Slack



III. longest path

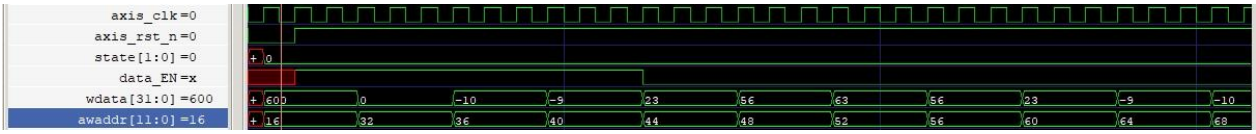



Tcl Console
Messages
Log
Reports
Design Runs
Timing
Intra-Clock Paths - axis_clk - Setup
Methodology Summ
Check Timing (326)
Intra-Clock Paths
axis_clk
Setup 0.586
Hold 0.070 r
Pulse Width
Inter-Clock Paths
Other Path Groups
Name
Slack
Levels
Routes
High Fanout
From
To
Total Delay
Logic Delay
Net Delay
Requirement
Source
Path 1 0.586 10 11 129 ctrl_tap/ffen_r_reg/C sm_tdata_r_reg[29]/D 11.309 8.452 2.857 12.0 axis_cl
Path 2 0.592 10 11 129 ctrl_tap/ffen_r_reg/C sm_tdata_r_reg[31]/D 11.303 8.446 2.857 12.0 axis_cl
Path 3 0.667 10 11 129 ctrl_tap/ffen_r_reg/C sm_tdata_r_reg[30]/D 11.228 8.371 2.857 12.0 axis_cl
Path 4 0.691 10 11 129 ctrl_tap/ffen_r_reg/C sm_tdata_r_reg[28]/D 11.204 8.347 2.857 12.0 axis_cl
Path 5 0.703 9 10 129 ctrl_tap/ffen_r_reg/C sm_tdata_r_reg[25]/D 11.192 8.335 2.857 12.0 axis_cl
Path 6 0.709 9 10 129 ctrl_tap/ffen_r_reg/C sm_tdata_r_reg[27]/D 11.186 8.329 2.857 12.0 axis_cl
Path 7 0.784 9 10 129 ctrl_tap/ffen_r_reg/C sm_tdata_r_reg[26]/D 11.111 8.254 2.857 12.0 axis_cl
Path 8 0.808 9 10 129 ctrl_tap/ffen_r_reg/C sm_tdata_r_reg[24]/D 11.087 8.230 2.857 12.0 axis_cl
Timing Summary - timing_1

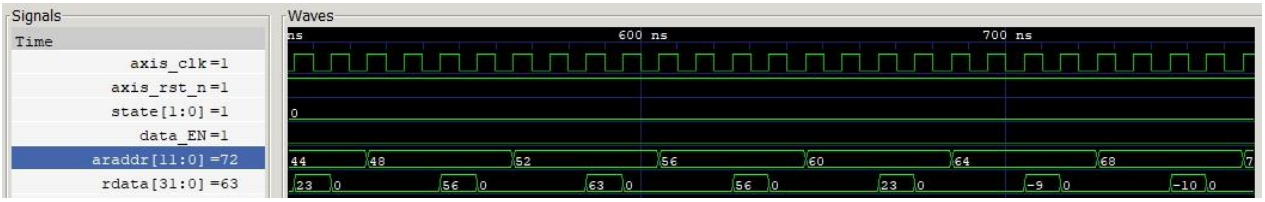

Tcl Console
Messages
Log
Reports
Design Runs
Timing
Intra-Clock Paths - axis_clk - Pulse Width
Clock Summary (1)
Methodology Summ
Check Timing (326)
Intra-Clock Paths
axis_clk
Setup 0.586
Hold 0.070 r
Pulse Width
Inter-Clock Paths
Check Type
Corner
Lib Pin
Reference Pin
Required
Actual
Slack
Location
Pin
High Pulse Width Fast FDRE/C n/a 0.500 6.000 5.500 FSM_onehot_state_reg[1]/C
High Pulse Width Fast FDRE/C n/a 0.500 6.000 5.500 FSM_onehot_state_reg[2]/C
High Pulse Width Fast FDRE/C n/a 0.500 6.000 5.500 ap_start_sig_reg/C
High Pulse Width Fast FDRE/C n/a 0.500 6.000 5.500 aready_r_reg/C
Low Pulse Width Fast FDSE/C n/a 0.500 6.000 5.500 FSM_onehot_state_reg[0]/C
Low Pulse Width Fast FDRE/C n/a 0.500 6.000 5.500 FSM_onehot_state_reg[1]/C
Low Pulse Width Fast FDRE/C n/a 0.500 6.000 5.500 FSM_onehot_state_reg[2]/C
Low Pulse Width Fast FDRE/C n/a 0.500 6.000 5.500 ap_start_sig_reg/C
Timing Summary - timing_1

● Simulation Waceform

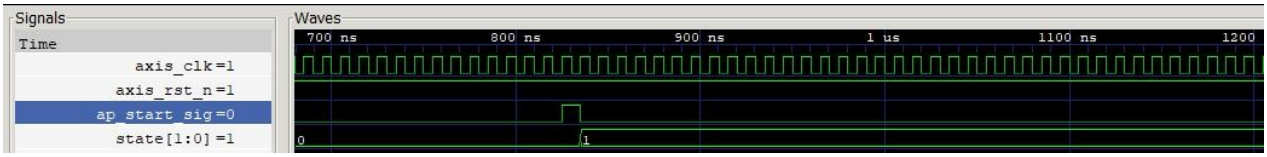
1.ap_idle =1 →state = 0，開始輸入 tap。



2.testbench 開始檢查 tap。

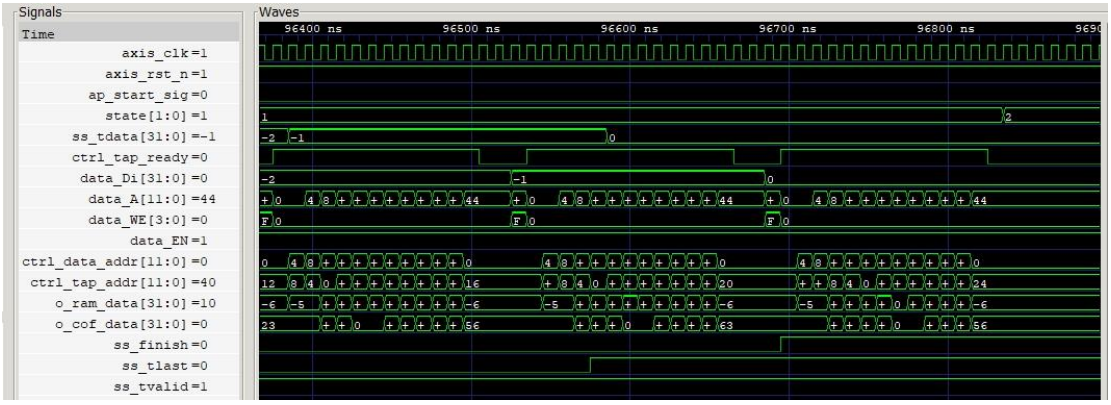


3.收到 ap_star_sig 訊號開始做運算，state = 2'b01。



4.state = 2'b01 後，會開始將 ss_tdata 寫入至 RAM，並且 ctrl_tap_ready = 1(下圖中紅色框框)，接著控制模組開始輸出對應的 tap_addr 與 data_addr，之後再 RAM 讀出數值做相乘。

5. 當 ss_tlast 和 sm_tlast 都為 1 後，ap_done=2 跳起來，故這邊會把 state = 2'b10。



- **observed & learned**

Lab3 的作業很具有挑戰性，我們花了許多天討論才討論出處理資料的方式，特別是把資料從 RAM 讀出來做相乘，這邊需要寫控制訊號來控制，我覺得也是最難的部分，所幸最後有想出辦法，但最後用到的 LUT 以及 FF 也還是很多，還要另找方法來解決，上課時有聽老師分享其他學校同學的方法，感覺可以試試看!!