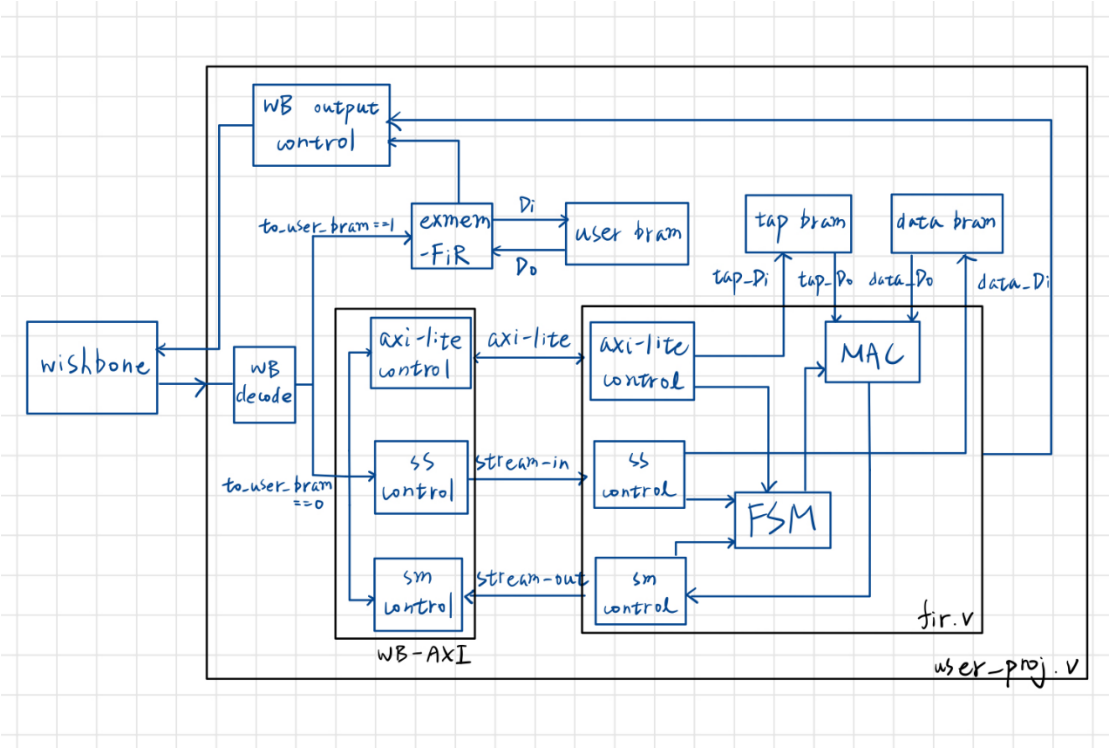
Lab 4-2 Report

組別：第 1 組　　姓名：游承緯 312580044　符顥瀚 312510154

1. Design block diagram – datapath, control-path



2. The interface protocol between firmware, user project and testbench

- Firmware 和 Testbench 之間的 interface protocol：
  - Given script to compile
    - `riscv32-unknown-elf-gcc -I../../firmware -o counter_la_fir.elf ..`
  - Transform .elf to .hex
    - `riscv32-unknown-elf-objcopy -O verilog counter_la_fir.elf counter_la_fir.hex`

```
spiflash #(
    .FILENAME("counter_la_fir.hex")
) spiflash (
    .csb(flash_csb),
    .clk(flash_clk),
    .io0(flash_io0),
    .io1(flash_io1),
    .io2(),         // not used
    .io3()          // not used
);
```

Spiflash：Firmware code 會以.hex 的形式 load 進 testbench 中的 Spiflash top

module

- Firmware 和 User Project 之間的 interface protocol：

  有 Wishbone 和 Logic Analyzer 兩種 interface protocol 可以使用

  Wishbone：

```
// Wishbone Slave ports (WB MI A)
input wb_clk_i,
input wb_rst_i,
input wbs_stb_i,
input wbs_cyc_i,
input wbs_we_i,
input [3:0] wbs_sel_i,
input [31:0] wbs_dat_i,
input [31:0] wbs_adr_i,
output reg wbs_ack_o,
output reg [31:0] wbs_dat_o,
```

  Wishbone 是我們 Lab 中主要使用的 interface protocol

  Logic Analyzer：

```
// Logic Analyzer Signals
input  [127:0] la_data_in,
output [127:0] la_data_out,
input  [127:0] la_oenb,
```

  有 128 個 input 以及 128 個 output，但在這次 Lab 中未使用。

- User Project、firmware 和 Testbench 之間的 interface protocol：

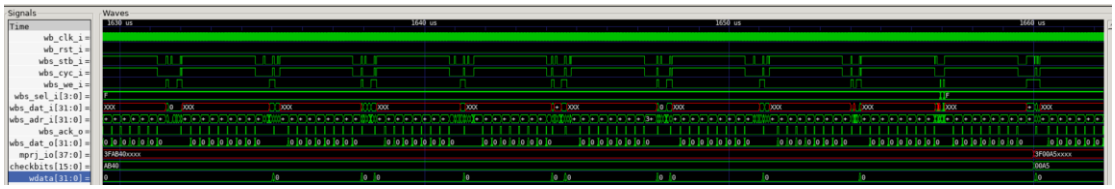
```
assign checkbits  = mprj_io[31:16];
```

```
wait(checkbits == 16'hAB40);
$display("LA Test 1 started");
```

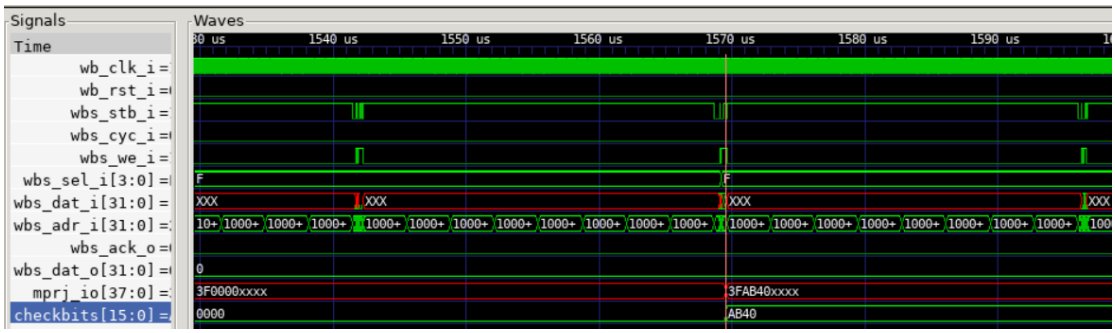
```
wait(checkbits == 16'hAB51);
$display("LA Test 2 passed");
```

  硬體跑的結果會透過 mprj_io 與 testbench 溝通。當 firmware code 執行到一個階段會去設定 mprj_io 的值，而 testbench 會去 check 它的值。

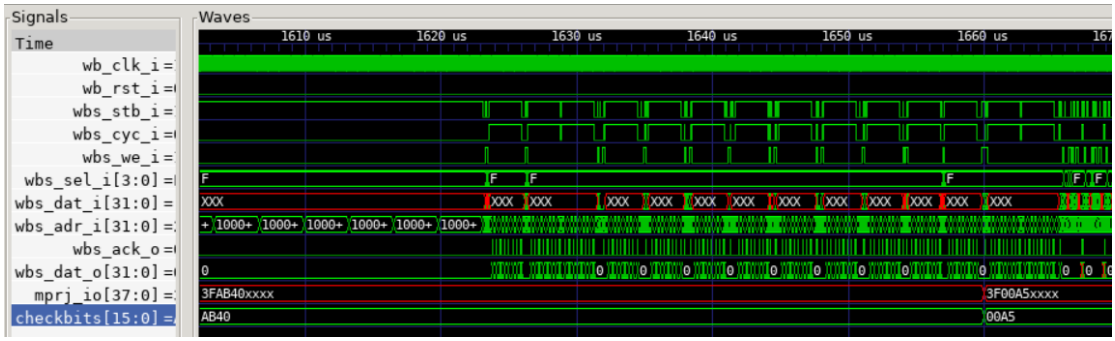
3. Waveform and analysis of the hardware/software behavior.



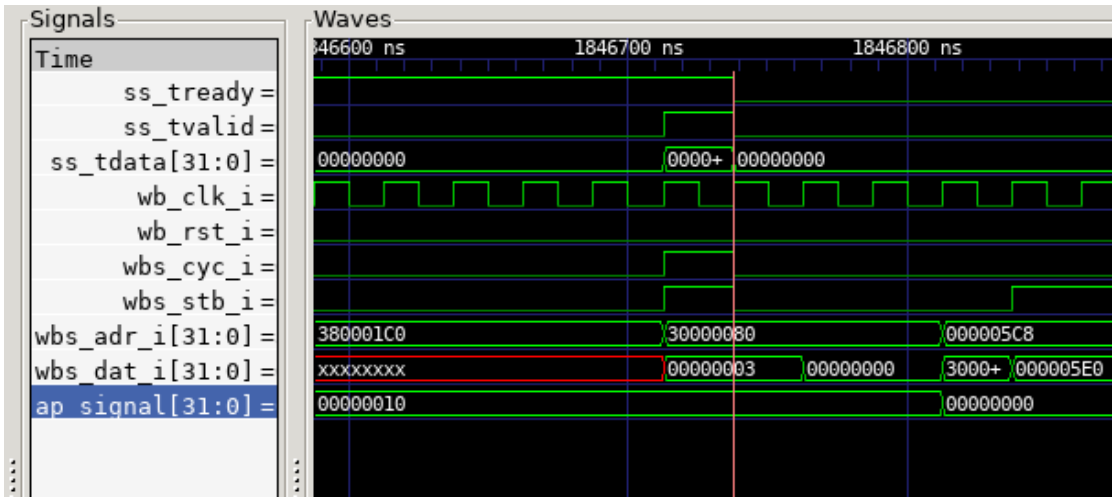
RISC-V program coeff, len



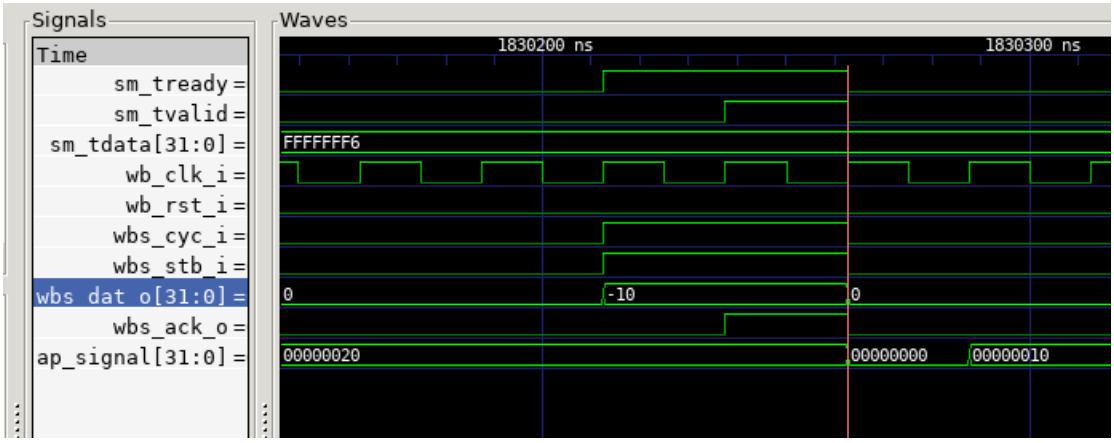
Checkbits == ‘hAB40，CPU 開始執行 firmware code



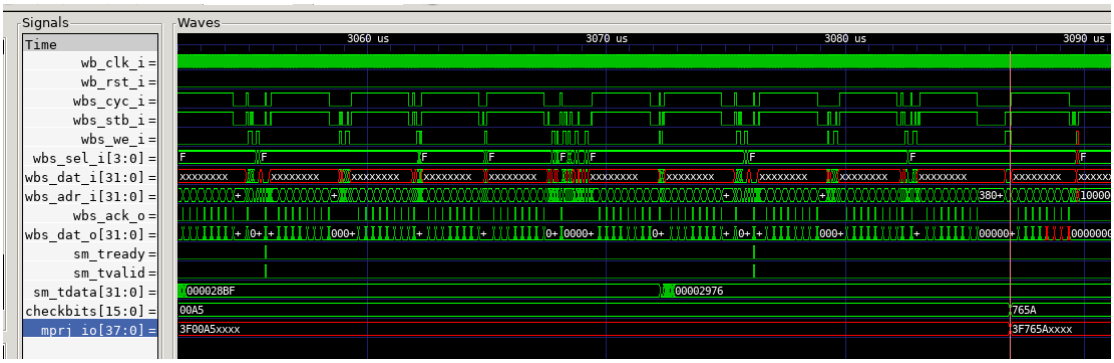
Checkbits == ‘hA5，testbench 收到 start mark，latency-timer start



RISC-V sends X[n] to FIR (ap_signal[4] == 1，wishbone 把 data 送進 fir。並把 ap_signal[4]歸 0，等 Yn 輸出才能拉起來為 1)



RISC-V received Y[n] from FIR(Y[n] is ready)

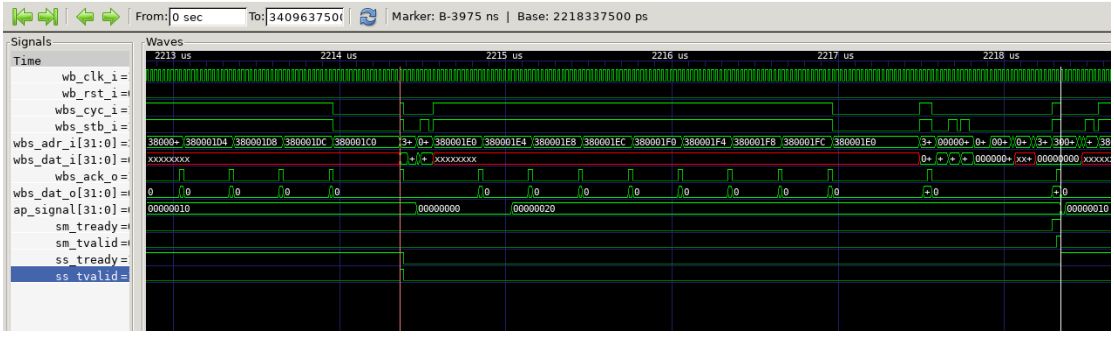


Checkbits[7:0] == ‘hA5，testbench 收到 end mark，record latency-timer

4. What is the FIR engine theoretical throughput, i.e. data rate? Actually measured throughput?

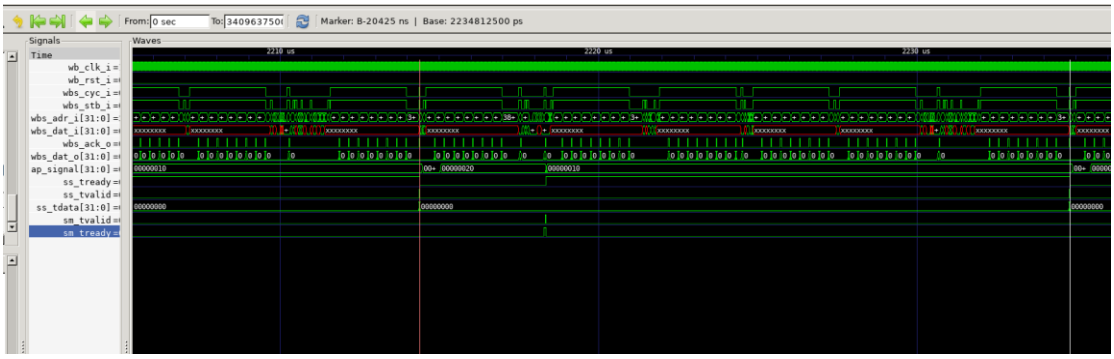
因為只有一個乘法器，且總共有 11 個 tap，理論上每 12 個 cycle 可以輸出一個 output，所以 throughput 為 32bits / 12cycles(300ns)

但由於我們在實作 FIR engine 時，沒有使用 pipeline 來處理 data，導致 Bram 使用效率低落。



所以實際的 throughput 如上圖，為 32bits / 159cycles(3975ns)

5. What is latency for firmware to feed data?



如上圖所示，input 進來的間隔是 20425ns

6. What techniques used to improve the throughput?

- 盡量讓 control 訊號有效縮短不同 protocol 之間轉換的 delay

7. Does bram12 give better performance, in what way?

我們這組在這次的 Lab 是使用兩個 bram11 來儲存 input 及 tap，所使用的面積會相較使用 bram12 儲存 input 來的小。使用兩個大小相同的 bram 也比較不會在設計時產生問題，以避免使用額外的控制訊號。

而若是我們是使用 bram12 來儲存 input，好處是我們可以使用多出來的位置，可能可以用來預先儲存 input，省下一點 cycle。相對的，這樣會增加設計的複雜性，需要額外的邏輯來確保正常運作。

我認為，這兩個方式的 performance 差距不大，但使用兩個相同大小的 bram 來設計會簡化許多，所以更傾向去使用 bram11。

8. Can you suggest other method to improve the performance?

- 使用多個乘加器來增加平行度
- 使用 read / write pointer 來實作可以將資料進行 pipeline 運算的 FIR engine
- 使用 Shift registers 來取代 Bram 來儲存 data，減少 Bram 所需的 delay

9. Syn Report

1. Slice Logic

Site Type	Used	Fixed	Prohibited	Available	Util%
Slice LUTs*	528	0	0	53200	0.99
LUT as Logic	464	0	0	53200	0.87
LUT as Memory	64	0	0	17400	0.37
LUT as Distributed RAM	64	0			
LUT as Shift Register	0	0			
Slice Registers	409	0	0	106400	0.38
Register as Flip Flop	377	0	0	106400	0.35
Register as Latch	32	0	0	106400	0.03
F7 Muxes	0	0	0	26600	0.00
F8 Muxes	0	0	0	13300	0.00

2. Memory

Site Type	Used	Fixed	Prohibited	Available	Util%
Block RAM Tile	1	0	0	140	0.71
RAMB36/FIFO*	1	0	0	140	0.71
RAMB36E1 only	1				
RAMB18	0	0	0	280	0.00

3. DSP

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

4. IO and GT Specific

Site Type	Used	Fixed	Prohibited	Available	Util%
Bonded IOB	301	0	0	125	240.80
Bonded IPADs	0	0	0	2	0.00
Bonded IOPADs	0	0	0	130	0.00
PHY_CONTROL	0	0	0	4	0.00
PHASER_REF	0	0	0	4	0.00
OUT_FIFO	0	0	0	16	0.00
IN_FIFO	0	0	0	16	0.00
IDELAYCTRL	0	0	0	4	0.00
IBUFDS	0	0	0	121	0.00
PHASER_OUT/PHASER_OUT_PHY	0	0	0	16	0.00
PHASER_IN/PHASER_IN_PHY	0	0	0	16	0.00
IDELAYE2/IDELAYE2_FINEDELAY	0	0	0	200	0.00
ILOGIC	0	0	0	125	0.00
OLOGIC	0	0	0	125	0.00

Design Timing Summary

WNS(ns)	TNS(ns)	TNS Failing Endpoints	TNS Total Endpoints	WHS(ns)	THS(ns)	THS Failing Endpoints	THS Total Endpoints	WPWS(ns)	TPWS(ns)	TPWS Failing Endpoints	TPWS Total Endpoints
1.261	0.000	0	957	0.079	0.000	0	957	6.250	0.000	0	444

All user specified timing constraints are met.