NTHU 112 Fall Semester

系統晶片設計

SOC Design Laboratory

Lab4-2 Caravel FIR

國立清華大學

NATIONAL TSING HUA UNIVERSITY

組別: 第 18 組

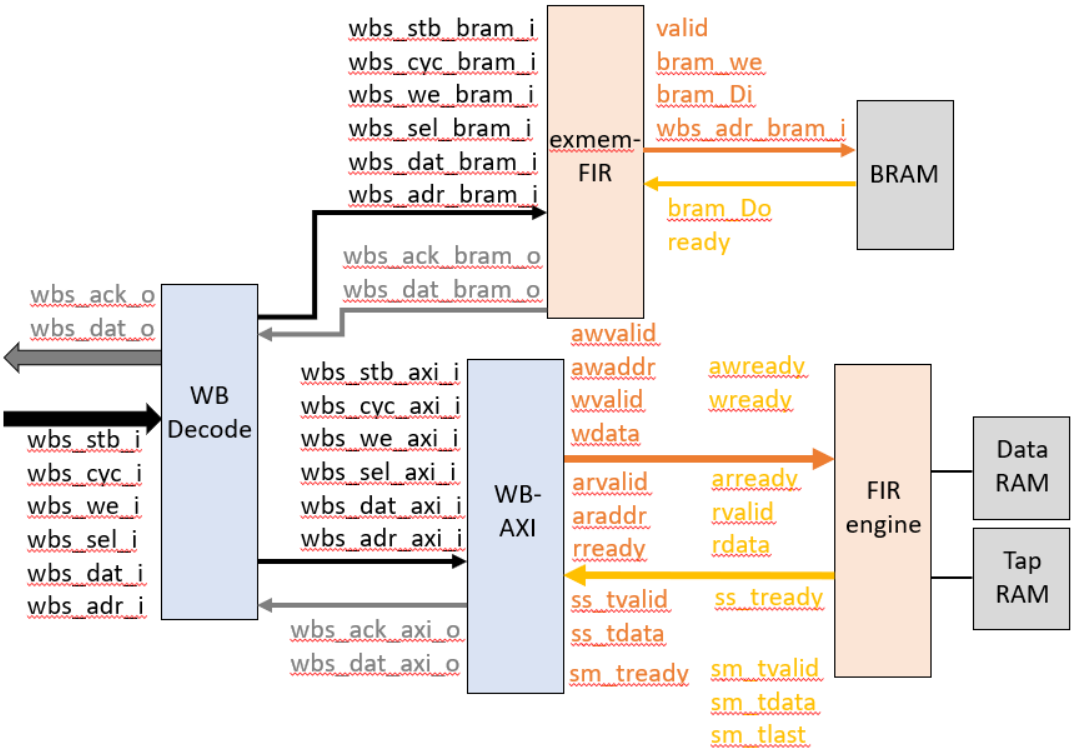
學號: 111061647, 111064537, 112061576

姓名: 盧人豪，林亭君，莊家政

1. **Github link:**
   https://github.com/lkl110137918218/SOC_lab_caravel_fir/tree/master

2. **Design block diagram datapath , control path**

   這次主要專注在藍色圖塊的部分。CPU 的 instruction code fetch 會從 0x3800_0000 的位置，要 program Verilog-FIR 就用 0x3000_0000 的位置，所以 decode 的方式就根據 wbs_adr_i[31:20]為 12'h380 or 12'h300 來決定要把 wishbone cycle 送到哪。



3. **The interface protocol between firmware, user project and testbench**

   User project 以及 firmware code 透過 mprj 來跟 testbench 做溝通。
   Firmware 跟 user project 之間的溝通則是透過 wishbone。
   一開始會把 firmware code compile 後的 hex file load 進 SPI Flash， 並將 inifir & fir 的部份放進 user bram 裡面，CPU 透過 management core 轉成 SPI Flash 的 cycle 來執行指令，然後透過 wishbone interface 對 user project 做動作。在執行時 CPU 會對 mprj pin 適時地做 configuration，來讓 testbench 確認是否正確。

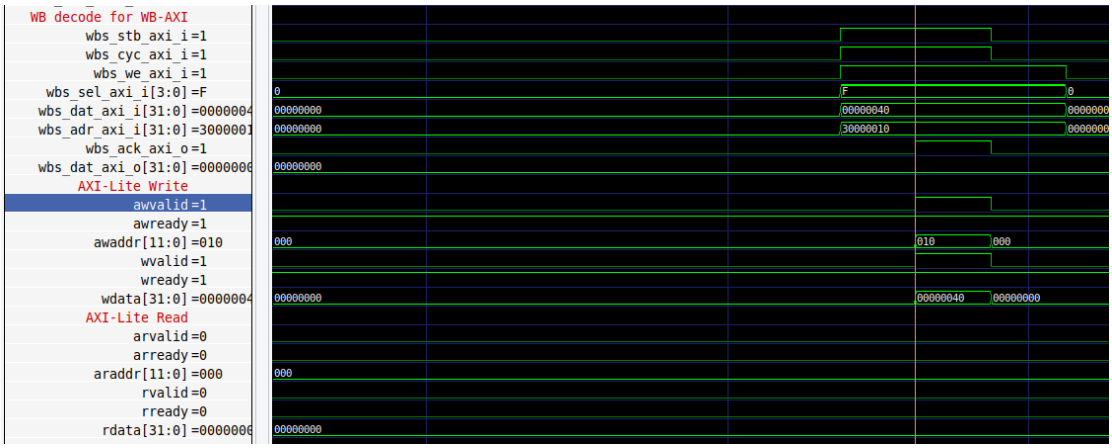
4. **Waveform and analysis of the hardware/software behavior**

   從下圖可以看到，當收到 wbs_adr_i[31:20]為 12'h300,decoded=2'b10，若為 12'h380 則 decoded=2'b01，在不同的 decoded 情況下分別讓 fir / user bram

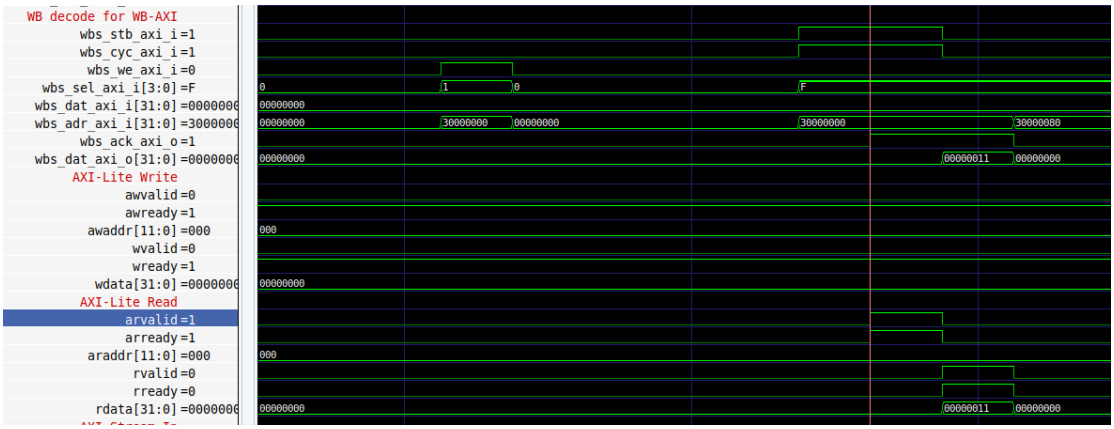
接收 or 控制 wishbone cycle (可看 WB decode for WB-AXI & WB decode for exmem-FIR 的訊號線)

Decode 過後，且 stb & cyc 信號都響應的情況下，便透過 wbs_we_axi_i 的訊號來控制要做 AXI-Lite read or write。

從下圖可以看到，由於 fir 的 awready & wready always 為 1 ，所以只要當 wbs_we_axi_i 響應時，就會將 awvalid & wavalid 拉為一，與此同時將 wbs_dat 及 wbs_adr 分別送入 fir 的 wdata 及 awaddr，就可以在一個 cycle 完成寫入動作

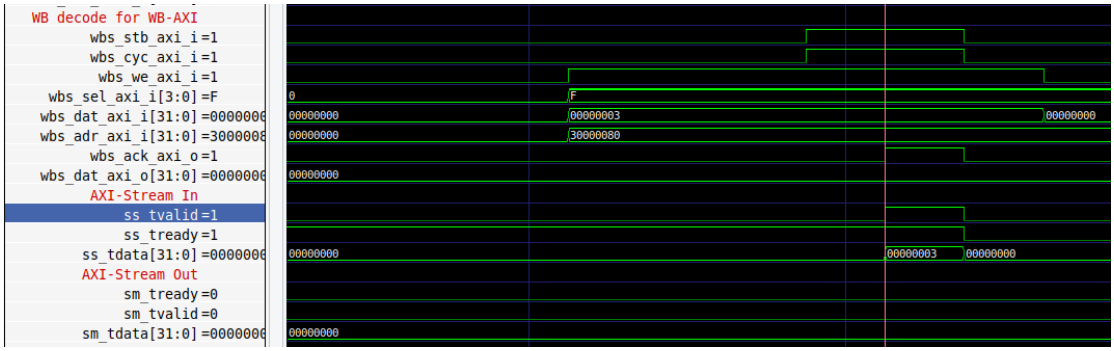


從下圖可以看到，由於 ar & r 信號有 dependency 關係，也就是一定要 arvalid & arready 都已經 assert 的前提下才能夠 assert rvalid signal，所以會至少需要兩個 cycle 才能把 data 讀出來，並且會將讀出來的 rdata 接上 wbs_dat_o

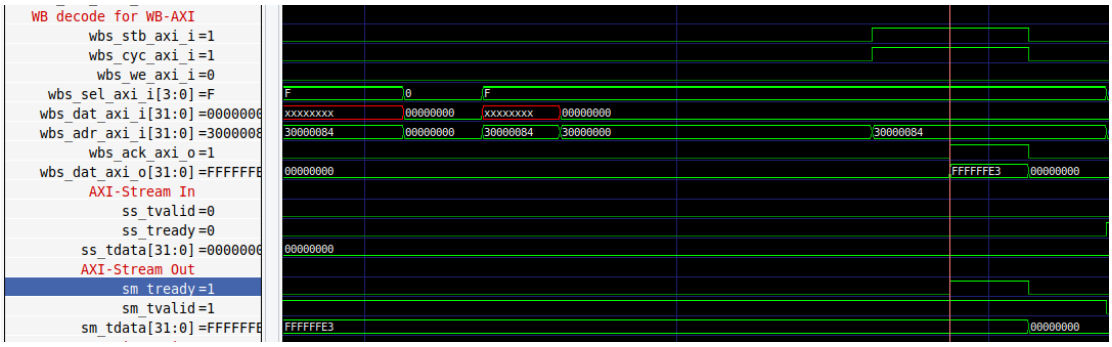


Decode 過後，且 stb & cyc 信號都響應的情況下，便透過 wbs_adr_axi_i[15:0] 為 16'h80 or 16'h84 決定是要進行 AXI-Stream in or AXI-Stream out。

若是要將 data 送進 fir，則在 ss_tvalid 拉為一的同時也會將 wbs_dat_i 給 ss_tdata。

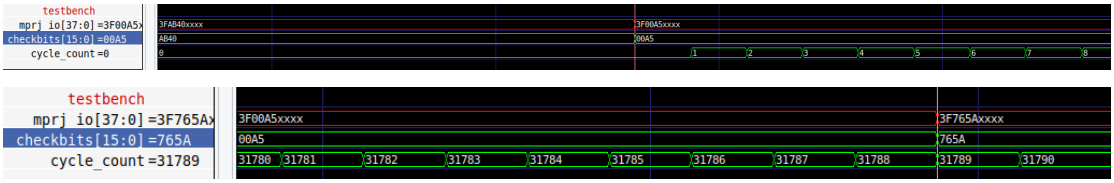


若是 fir 要將 data 送出，則在 ss_tready 拉為一的同時也會將計算完的 sm_tdata 送給 wbs_dat_o。



另外，可以看到當 RISC V outputs 出 Start Mark ('hA5) 在 mprj [23:16] 時，會在 testbench 開始做 latency count，並在收到 765A 後(最後一筆 Y[7:0]

output 接到 mprj [31:24] & EndMark ('h5A)接到 mprj [23:16]),停止算 latency



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

理論上，要計算出一個 output data，需要 11 cycles，所以 data rate 應該要是 1/11 = 0.091 data/cycle。

實際上，從下圖模擬結果可以看到，要計算完 64 筆 output data，總共會需要 31790 cycles，所以 data rate 為 64/31790 = 0.00201321 data/cycle。

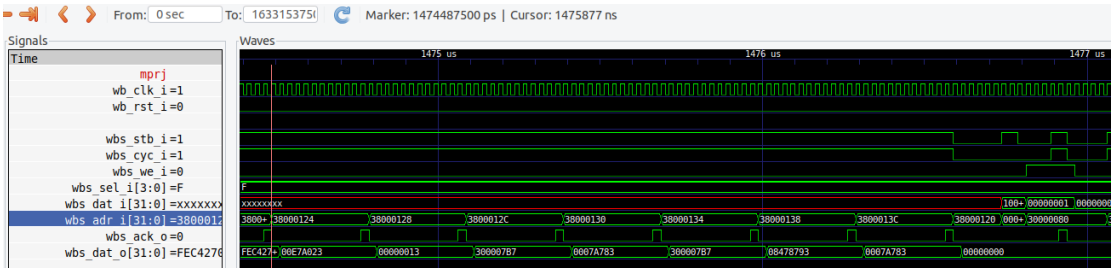
```
ubuntu@ubuntu2004:~/course-lab_4/lab4_2-git/testbench/counter_la_fir$ source run_sim
Reading counter_la_fir.hex
counter_la_fir.hex loaded into memory
Memory 5 bytes = 0x6f 0x00 0x00 0x0b 0x13
VCD info: dumpfile counter_la_fir.vcd opened for output.
LA Test 1 started
checkbits = 765a
finsih first iteration test
latency of 1st itertaion =       31790 clock cycles
checkbits = 765a
finsih second iteration test
latency of 2nd itertaion =       31790 clock cycles
checkbits = 765a
finsih third iteration test
latency of 3rd itertaion =       31790 clock cycles
latency of total itertaion =       95370 clock cycles
LA Test 2 passed
```

6. What is latency for firmware to feed data?

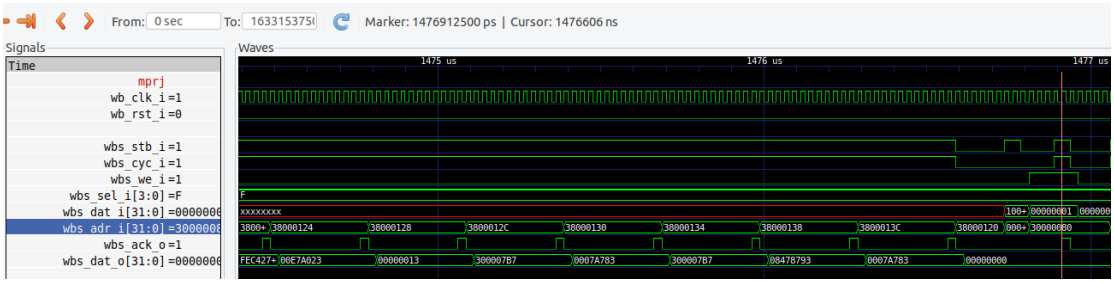
在 firmware code 下了送資料的指令開始，直到 fir 收到該筆資料停止，計算總共經過的 cycle count，這邊以送第二筆 Xn 來看，經過了

(1476912500 – 1474487500) / 25000 = 97 cycles

Firmware code 下了送第一筆資料的時間點:



Fir 實際收到第一筆資料的時間點:



7. What techniques used to improve the throughput?

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

我們並沒有使用 bram12，但或許可以利用多出來的一個位置來存放每次計算完的 output 結果，這樣就可以馬上開始算下一筆 data，而不會因為在等待 output 指令的關係導致運算資源 idle 。

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

可以考慮使用多個乘法、加法器來做平行化的運算，或是改善 firmware code 的寫法，縮短等待 output 指令的時間。

8. Metrics: of clock (latency timer) * clock_period * gate resource

- # of clock: 31790 cycles
- clock_period - the longest path in static timing report: 10.261 ns
- gate resource - # of (LUT + FF): 686

Metrics = 0.22377

9. Timing report

Periods = 11 ns

```
------------------------------------------------------------------
| Clock Summary
| -------------
------------------------------------------------------------------

Clock     Waveform(ns)       Period(ns)      Frequency(MHz)
-----     ------------       ----------      --------------
wb_clk_i  {0.000 5.500}      11.000          90.909
```

Timing slack is MET，no timing violation

```
------------------------------------------------------------------------------------------------
| Timing Details
| --------------
------------------------------------------------------------------------------------------------


------------------------------------------------------------------------------------------------
From Clock:  wb_clk_i
  To Clock:  wb_clk_i

Setup :            0  Failing Endpoints,  Worst Slack        0.603ns,  Total Violation        0.000ns
Hold  :            0  Failing Endpoints,  Worst Slack        0.144ns,  Total Violation        0.000ns
PW    :            0  Failing Endpoints,  Worst Slack        4.250ns,  Total Violation        0.000ns
------------------------------------------------------------------------------------------------
```

10. Synthesis report

將 user_proj_example.counter.v & bram.v & fir.v & bram11.v 拿下去做合成

1. Slice Logic

Site Type	Used	Fixed	Prohibited	Available	Util%
Slice LUTs*	507	0	0	53200	0.95
LUT as Logic	443	0	0	53200	0.83
LUT as Memory	64	0	0	17400	0.37
LUT as Distributed RAM	64	0			
LUT as Shift Register	0	0			
Slice Registers	179	0	0	106400	0.17
Register as Flip Flop	179	0	0	106400	0.17
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

2. Memory

Site Type	Used	Fixed	Prohibited	Available	Util%
Block RAM Tile	4	0	0	140	2.86
RAMB36/FIFO*	4	0	0	140	2.86
RAMB36E1 only	4				
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				

7. Primitives

Ref Name	Used	Functional Category
FDRE	177	Flop & Latch
OBUF	176	IO
LUT6	145	LUT
LUT2	134	LUT
LUT5	82	LUT
IBUF	71	IO
RAMS32	64	Distributed Memory
OBUFT	64	IO
LUT4	57	LUT
LUT3	36	LUT
LUT1	31	LUT
CARRY4	26	CarryLogic
RAMB36E1	4	Block Memory
DSP48E1	3	Block Arithmetic
FDSE	2	Flop & Latch
BUFG	1	Clock