NYCU-ECE DCS-2022

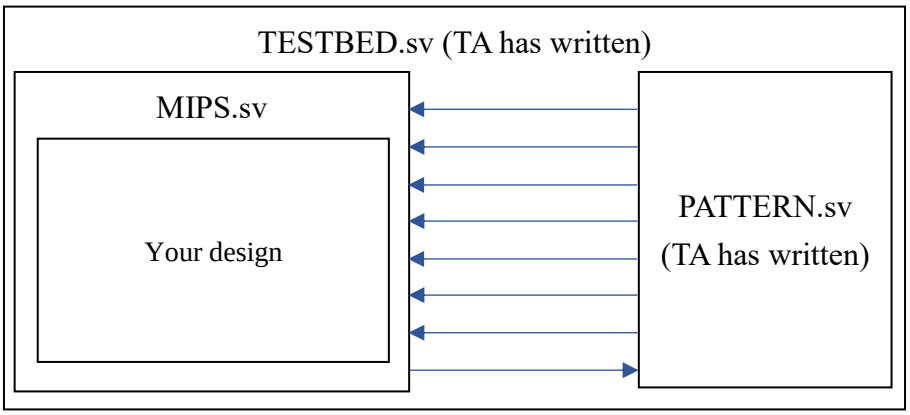
HW04

Design: MIPS CPU with Pipeline

資料準備

1. 從 TA 目錄資料夾解壓縮:
   % tar -xvf ~dcsta01/ HW04.tar
2. 解壓縮資料夾 hw04 包含以下:
   a. 00_TESTBED/
   b. 01_RTL/
   c. 02_SYN/
   d. 03_GATE/
   e. 09_UPLOAD/

Block Diagram



設計描述

此次作業目的是設計一個CPU執行簡單運算並使用pipeline，可以連續執行序列的Instruction。MIPS是一種指令集，內含有許多指令，此次作業以MIPS指令集做稍微修改。

Type	Instruction 32 bits					
R	Opcode (6 bits)	Rs (5 bits)	Rt (5 bits)	Rd (5 bits)	Shamt (5 bits)	Funct (6 bits)
	Opcode	funct		operation		
	000000	100000		Rs + Rt		
		100100		Rs and Rt		
		100101		Rs or Rt		
		100111		Rs nor Rt		
		000000		Rt 向左 shift shamt bits		
		000010		Rt 向右 shift shamt bits		
Type	Instruction 32 bits					
I	Opcode (6 bits)	Rs (5 bits)	Rt (5 bits)	Immediate (16 bits)		
	Opcode	operation				
	001000	Rs + imm (16 bits)				

rs: source register address, rt: target register address,

rd: destination register address, imm :value

Instruction會給予不在列表中的指令，當遇到這種情況時，out_1，out_2，out_3，out_4皆設置為零，並且將intruction_fail 設為1。
除了Instruction外，還有兩個input : in_valid、output_reg
in_valid為high時代表instruction與output_reg開始給值，design要開始取值。

output_reg代表最後output的四個signal需要取的是哪四個暫存器，給的值是address，並且分別將暫存器的value給out_1、out_2、out_3、out_4，如下表所示。

[19:0]output_reg	[19:15]	[14:10]	[9:5]	[4:0]
	out_4	out_3	out_2	out_1

暫存器需自行宣告，各暫存器的位址如下表。

Address(5 bits)	Value(32 bits) 初始值皆為零
10001	0
10010	0

01000	0
10111	0
11111	0
10000	0

R、I type有寫回功能，分別用Rd/Rt表示寫回位址，須將值存起來進行下次計算。

以下為一些指令例子：

1. Instruction : 001000-10001-10010-0000000000000100

   Opcode-rs-rt-imm

   Addi instruction, reg(10001) = 0，imm = 4, reg(10010) = 0 + 4 = 4

2. Instruction : 001100-10001-10010-0000000000000100

   NO opcode 001100 -> instruction fail !

3. Instruction : 000000-10001-10010-10111-00001-000000

   Opcode-rs-rt-rd-shamt-funct

   Sll instruction, reg(10010) = 4, shamt = 1, reg(10111) = 8

可以從以上範例來得知，當第一行指令執行完後，reg(10010)值變為4，而第三行指令又拿取reg(10010)的值，這次會延續第一行的結果拿到4。

這次作業助教只會提供十筆測資，同學們需要自行產生測資以驗證自己的設計，可以更改input.txt/output.txt的檔案或改寫PATTERN.sv，demo時助教會使用不只十筆測資。

input.txt 範例

```
10
00100010001100100000000000000100 01000100011011101000
00110000000000000000000000100110 11111100011111110001
00000010001100101011100001000000 01000100001111110000
00000001000100011000011101000010 11111100011111110001
00100010111100000000000101011011 10010100001001010111
11000000000000000000000000100110 11111100011111110001
00000010001100001111100101100000 10001100001001010111
11000000000000000000000000100110 11111100011111110001
00000011111100101000110000100101 01000100011011111111
00000010000111111111100110100111 10111101111000010111
```

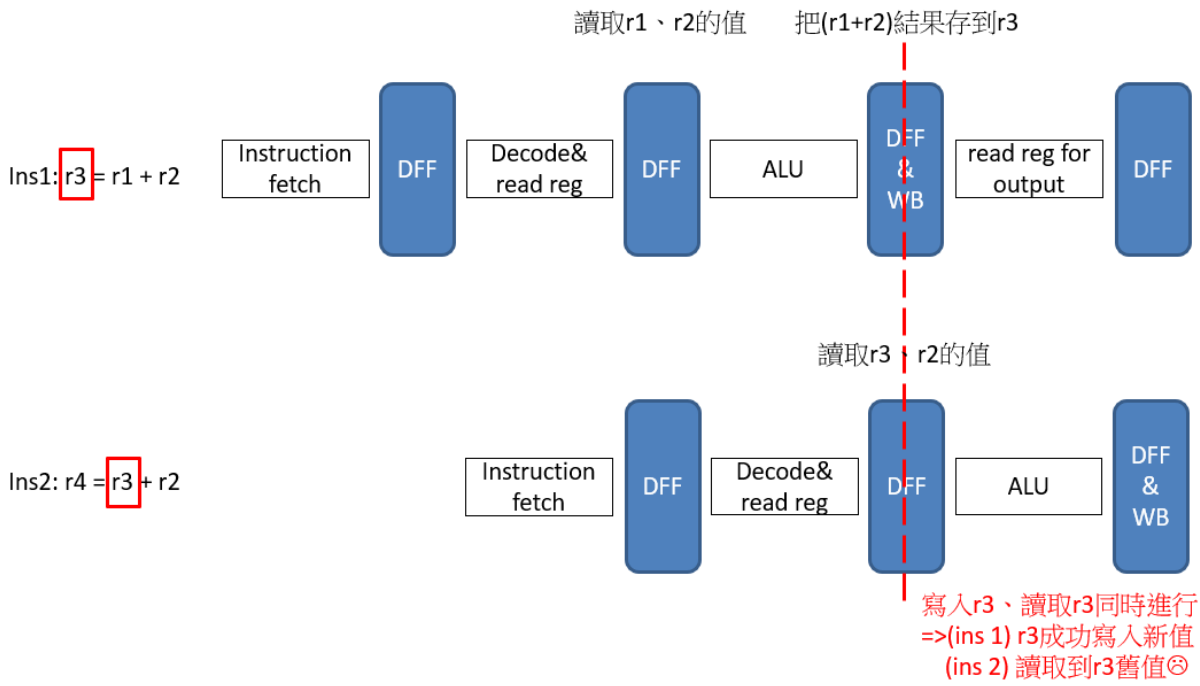
第一行是測資數量，第二行開始依序為instruction、output_reg

output.txt範例

1	0	0	0	0	0
2	1	0	0	0	0
3	0	0	0	0	0
4	0	0	0	0	0
5	0	8	4	703	4
6	1	0	0	0	0
7	0	8	4	703	0
8	1	0	0	0	0
9	0	703	8	0	0
10	0	8	703	8	8

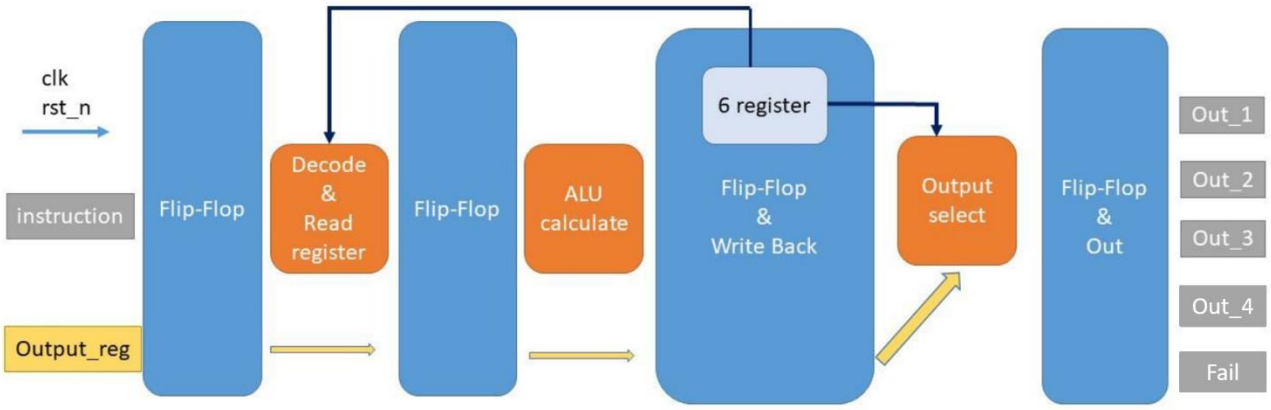
順序為：instruction_fail、out_1、out_2、out_3、out_4

※這次的design架構會有data hazard RAW(read after write)的問題，也就是上一個指令還未成功寫入reg，下一個指令先讀取了舊的值，助教產生的測資會避開造成data hazard指令，同學們自己產生測資時需要特別注意。以下為範例：



想知道更詳細的可以參考連結：https://king0980692.medium.com/computer-architecture-cheat-sheet-pipeline-hazard-ee27d0d66e89

下圖是結構圖，依照結構圖寫出自己的design基本上就能pass(pipeline級數需相同)。



Inputs

Signal name	Number of bit	Description
clk	1 bit	Clock
rst_n	1 bit	Asynchronous active-low reset
in_valid	1 bit	When getting high, means start giving instruction
instruction	32 bits	In_valid 為 1 的時候給值，並且一個 cycle 一個 instrcution
output_reg	20 bits	指定輸出的四個 output signal 分別為哪四個 register

Outputs

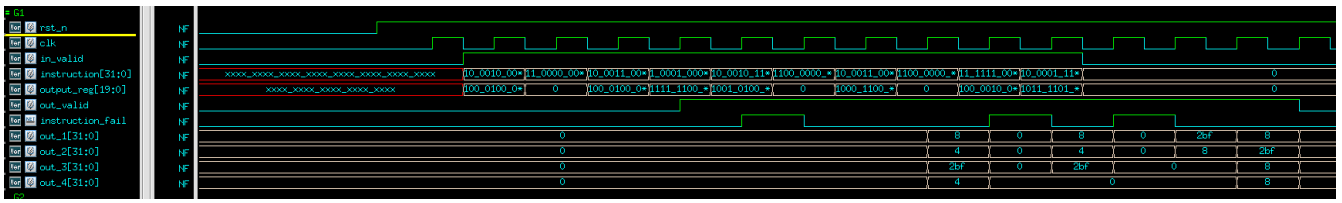
Signal name	Number of bit	Description
out_valid	1 bit	輸出有值的時候為 high
out_1	32 bits	Address 是 output_reg[4:0]的暫存器數值，instruction 無效時為零。
out_2	32 bits	Address 是 output_reg[9:5]的暫存器數值，instruction 無效時為零。
out_3	32 bits	Address 是 output_reg[14:10]的暫存器數值，instruction 無效時為零。
out_4	32 bits	Address 是 output_reg[19:15]的暫存器數值，instruction 無效時為零。
instruction_fail	1 bit	instruction 無效時為 1，有效為 0。

Specifications

1. Top module name: **MIPS**(File name : **MIPS.sv**)

2. 在非同步負準位 reset 後，所有的 output 訊號必須全部歸零。
3. Output 要在 Input 後的 10 cycles 內輸出。
4. 所有 Output 訊號要在輸出結束後全部歸零。
5. 02_SYN result 不行有 error 且不能有任何 latch。
6. Clock period 5 ns。
7. Input delay = 0.5 * clock period; Output delay = 0.5 * clock period
8. Separate your combination and sequential blocks!

Example waveform



上傳檔案

1. Code使用09_upload上傳。
2. report_dcsxx.pdf, xx is your server account. 上傳至new E3。
3. 1 demo請在 4/28 15:30 pm 上課之前上傳。
4. 2 demo請在 5/05 15:30 pm 上課之前上傳。

Grading policy

1. Pass the RTL& Synthesis simulation. 60%
2. Area 30%
3. Report 10%
4. Combinational、sequential Logic沒有分開寫 -5%

Note

Template folders and reference commands:

1. 01_RTL/ (RTL simulation) → **./01_run**
2. 02_SYN/ (synthesis) → **./01_run_dc**
3. 03_GATE/ (gate-level simulation) → **./01_run**
4. 09_UPLOAD/ (upload) → **./09_upload**

報告請簡單且重點撰寫，不超過兩頁A4，並包括以下內容

1. 描述你的設計方法，包含但不限於如何加速(減少critical path)或降低面積。
2. 基於以上，畫出你的架構圖(Block diagram)
3. 心得報告，不侷限於此次作業，對於作業或上課內容都可以寫下。
4. 遇到的困難與如何解決。