NCTU-EE DCS-2019

HW02

Design: MIPS CPU with Pipeline

資料準備

1. 從 TA 目錄資料夾解壓縮

   % tar -xvf ~dcsta01/HW02.tar

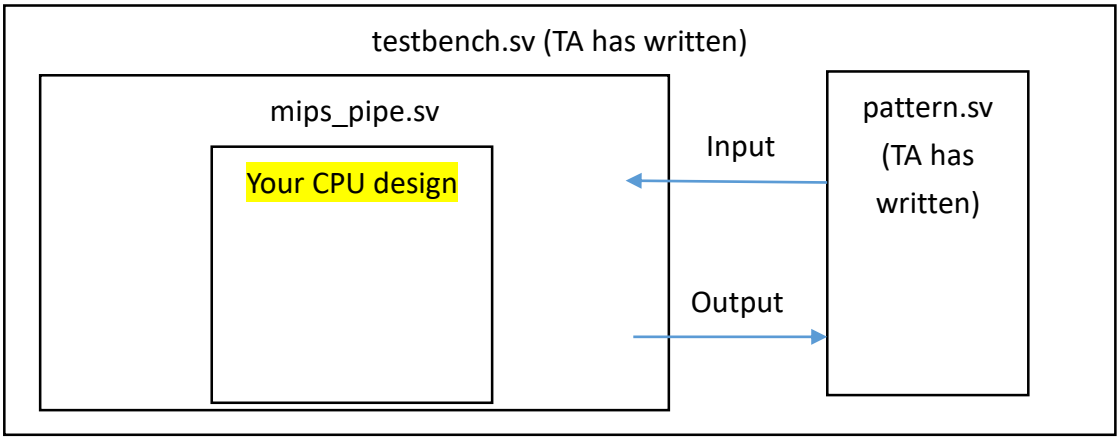
2. 解壓縮資料夾 HW02 包含以下：

   A. 00_TESTBED/

   B. 01_RTL/

   C. 02_SYN/

Block Diagram



設計描述

此次作業目的是設計一個簡單的 CPU 執行簡單運算並且擁有 pipeline，可以連續吃序列的 Instruction。關於 CPU 內擁有的指令與 HW01 相似，可以看以下表格。指令的相關意義不再說明，請回顧 HW01。

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 shift 向左 shamt bits		
		000010		rt shift 向右 shamt bits		
Type	Instruction 32 bits					
I	Opcode (6 bits)	rs (5 bits)	rt (5 bits)	immediate (16 bits)		
	opcode	Operation				
	001000	rs + imm (16bits)				

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

rd: destination register address, imm :value

這次的 Instruction 會給予不在列表中的 instruction，當遇到這種情況時，out_1，out_2，out_3 皆為零，並且將 intruction_fail 設為 1。

除了 Instruction 外，還有兩個 input 需要注意:in_valid、output_reg。

in_valid 為 high 時代表 instruction 與 output_reg 開始吐值，design 要開始取值。

output_reg 代表最後 output 的三個 signal 需要取的是哪三個暫存器，給的值是 address，並且分別將暫存器的 value 給 out_1、out_2、out_3，如下表所示。

Bit	[14:10]	[9:5]	[4:0]
[14:0]output_reg	暫存器的 address	暫存器的 address	暫存器的 address
output signal	out_3	out_2	out_1

這次作業與 HW01 不同的部分，暫存器不用使用助教給的模組，而是自行在 design 內寫出 6 個暫存器，各自暫存器的位址如下表。

六個暫存器位置如下：

Address(5 bits)	**Value(32 bits) 初始皆為 0**
10001	0
10010	0
01000	0
10111	0
11111	0
10000	0

與 HW01 還有一點不同是必須有「寫回」功能，Rd address 用來將運算結果寫回該位置，並且將值儲存起來下次繼續計算。

以下為一些指令的例子：

1. Instruction : 001000-10001-10010-0000000000000100

   Opcode - rs - rt - imm

   From opcode we can know this is the “addi” instruction.

   Note: “addi” instruction destination is rt, not rd. So, rt values is output.

   It means register(10001) = 0, imm = 4 , rt(10010) = 0 + 4 = 4.

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

   Opcode- rs - rt- rd - imm- func

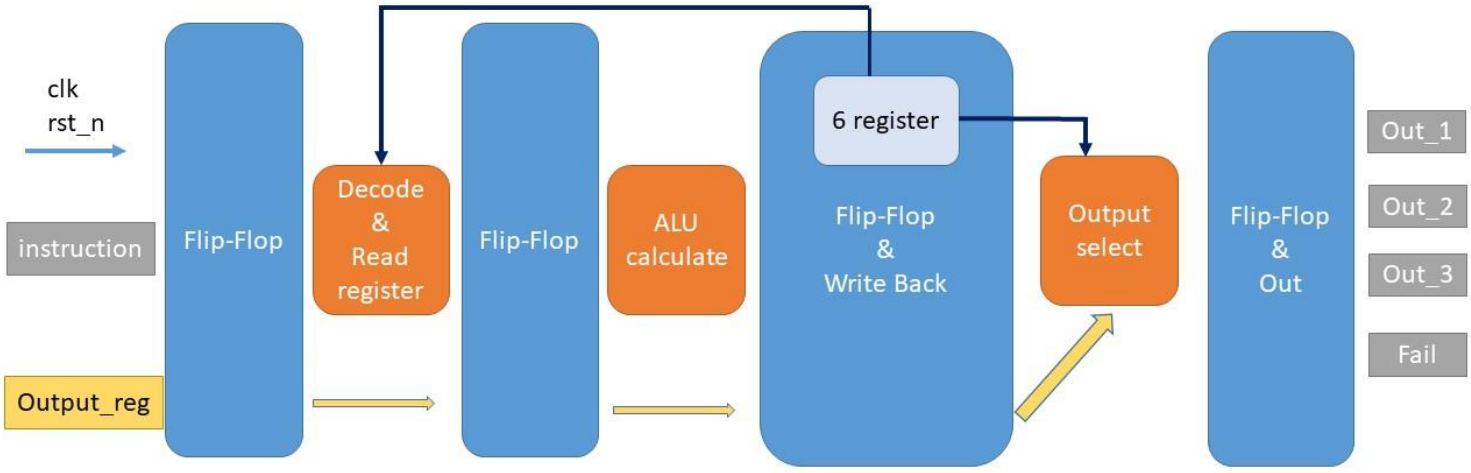
   From opcode & funct we can know this is the “sll” instruction.

   Note: “sll” instruction 是必須移動 rt 的資料而不是 rs.

   It means register(10010) = 4 shift 1-bit left and rd = 8

   可以從以上兩行指令範例來得知，當第一行指令執行完後，reg(10010)值變為 4，而第二行指令又拿取 reg(10010)的值，這次會延續第一行的結果拿到 4。

下圖附上結構圖，依照結構圖寫出自己的 design 基本上就能順利 pass(pipeline 級數需相同)。



Input

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**	當 in_valid 為 high 時代表有值，並且一個 cycle 一個 instruction
output_reg	**15 bits**	指定輸出的三個 output signal 分別為哪三個 register

Output

Signal name	Numb	Description
out_valid	**1 bits**	當做完運算之後，隨著 out_1、out_2、out_3 有值時給予 high
out_1	**32 bit**	Register 的 value 依照 output_reg[4:0]所指定的 register。當 instruction 無效時為 0。
out_2	**32 bit**	Register 的 value 依照 output_reg[9:5]所指定的 register。當 instruction 無效時為 0。
out_3	**32 bit**	Register 的 value 依照 output_reg[14:10]所指定的 register。當 instruction 無效時為 0。
instruction_fail	**1 bit**	當 instruction 無效時為 1，有效時為 0。

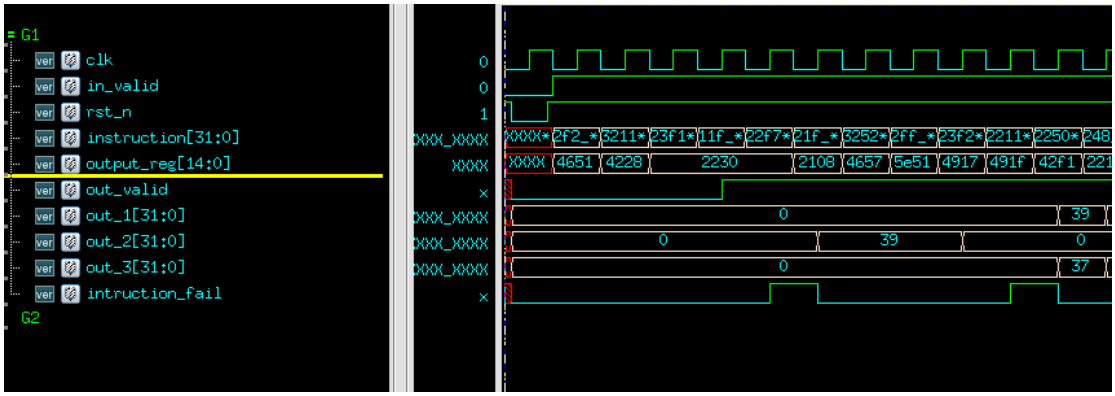
以上的 output 當 pattern 吐完 input、計算完後皆須歸為 0。

Specification

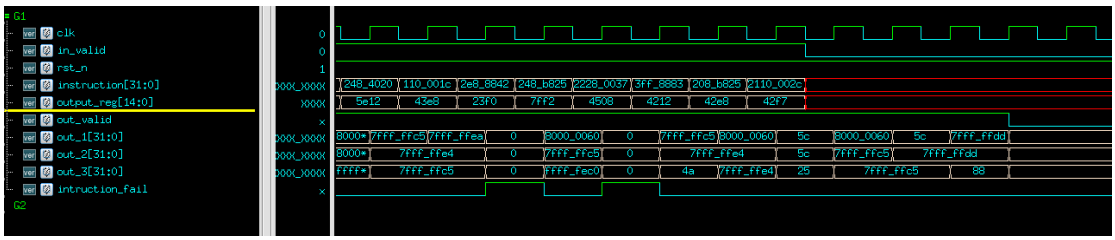
1. Top module name : mips_pipe(File name: mips_pipe.sv)
2. 請用 System verilog 完成你的作業。
3. 請盡量使用助教提示的架構圖。
4. 02_SYN result 不行有 error 且不能有 latchs。

Example Waveform

Pattern 一開始



Pattern 結束



上傳檔案

1. mips_pipe_dcsxx.sv (如果需要更新，請直接替換原始舊檔案)
2. report_dcsxx.pdf, **xx is your server account.**
3. 請 **4/5 9:00 am 之前上傳**

Grading Policy

1. Pass the RTL & Synthesis simulation. 70%
2. Area 15%
3. Report. 15%

Note

Template folders and reference commands:

1. 01_RTL/ (RTL simulation) ./01_run
2. 02_SYN/ (Synthesis) ./01_run_dc

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。