113 學年第一學期計算機系統設計期末考 A 卷

- 請完成下面三道題目。

  一.組合語言撰寫—找最大值 (40%)

  二.XCH16.R rd, rs1 (40%)

  三.CNZDECJ rd, rs1, imm (40%)

- 題目與架構如下面說明，實現完畢請比對波形圖，評分標準在本卷最後面。

分數分配:

1. 總分 120 分，超過 100 分以 100 分計算
2. 越早交越高分
3. 第一題完成後可先舉手 demo，第二第三題則一起 demo
4. 組合語言程式執行結果正確 40%
5. XCH16.R 指令執行正確 40%
6. CNZDECJ 指令執行正確 40%

若結果無誤或是有任何提問請舉手

一.組合語言撰寫—找最大值

1. 說明：

i. 如圖 1 所示，在記憶體位址 0x5 到 0xE 的地方共存有 **10 筆 8 位元的有號數**，請依據圖 2 中的演算法撰寫一個**組合語言**程式找出最大值 (0x6E) 並將其存至 x31 暫存器中。

ii. 請注意期末考和小考新增的**自定義指令是無法使用**的，因為組譯器不支援這些指令。

iii. 部分組合語言已提供在圖 3，請同學在撰寫完畢後使用 Venus 套件組譯成機器碼，透過 ModelSim 觀察程式執行的結果。

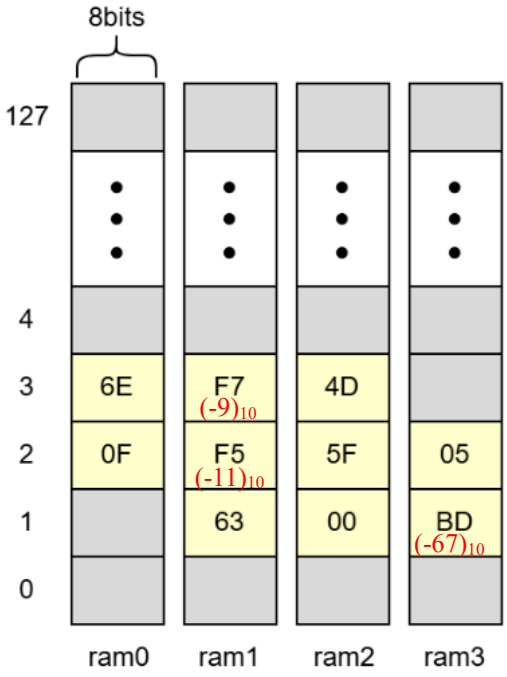


圖 1. 記憶體內的資料

```c
// 使用x31暫存器儲存最大值
register int x31 asm("x31");
// 初始化最大值為第一筆資料
x31 = memory[0x5];

// 找出最大值
for (int i = 0x5; i < 10; i++) {
    if (memory[i] > x31) {
        // 更新最大值
        x31 = memory[i];
    }
}
```

圖 2. C 語言—找最大

```
init:
    li x2, 5
    li x3, 0x0fbd0063
    sw x3, 0(x2)
    li x3, 0x6e055ff5
    sw x3, 4(x2)
    li x3, 0x00004df7
    sw x3, 8(x2)

#-----開始撰寫您的組合語言-----
```

圖 3. 部分組合語言

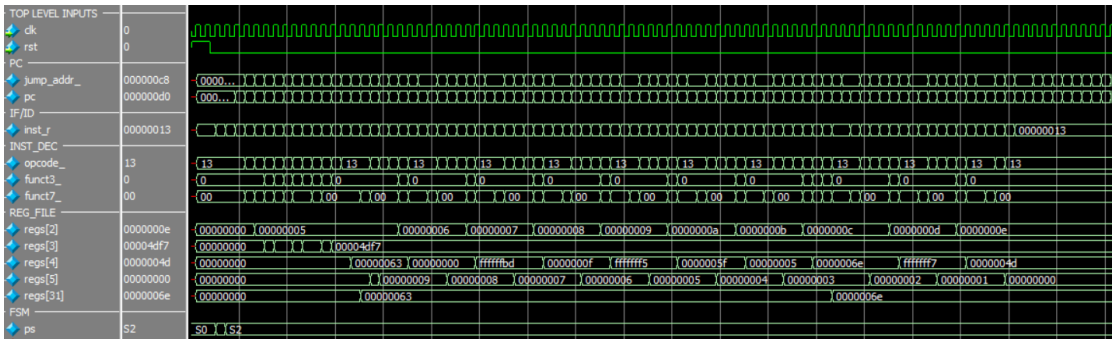


圖 4. 模擬結果波形(參考)

二. XCH16.R rd, rs1

1. 指令格式 :

Immediate value	Source registers 1	Funct3	Destination registers	Opcode	Instruction
12'b0	rs1	000	rd	1011011	XCH16.R
31 20	19 15	14 12	11 7	6 0	

2. 指令說明 : x[rd] ← { x[rs1][15:0], x[rs1][31:16] }

   XCH16.R (Exchange 16-bit Halves - Register) 指令的功能是將暫存器 rs1 中的高 16 位與低 16 位互換，並將交換後的結果寫入暫存器 rd。

   ※ 此指令的立即值解碼方式與立即值定址指令相同。

三. CNZDECJ rd, rs1, imm

1. 指令格式 :

Immediate value	Source registers 2	Source registers 1	Funct3	Destination registers	Opcode	Instruction
imm[6:0]	00000	rs1	010	rd	1111011	CNZDECJ
31 25	24 20	19 15	14 12	11 7	6 0	

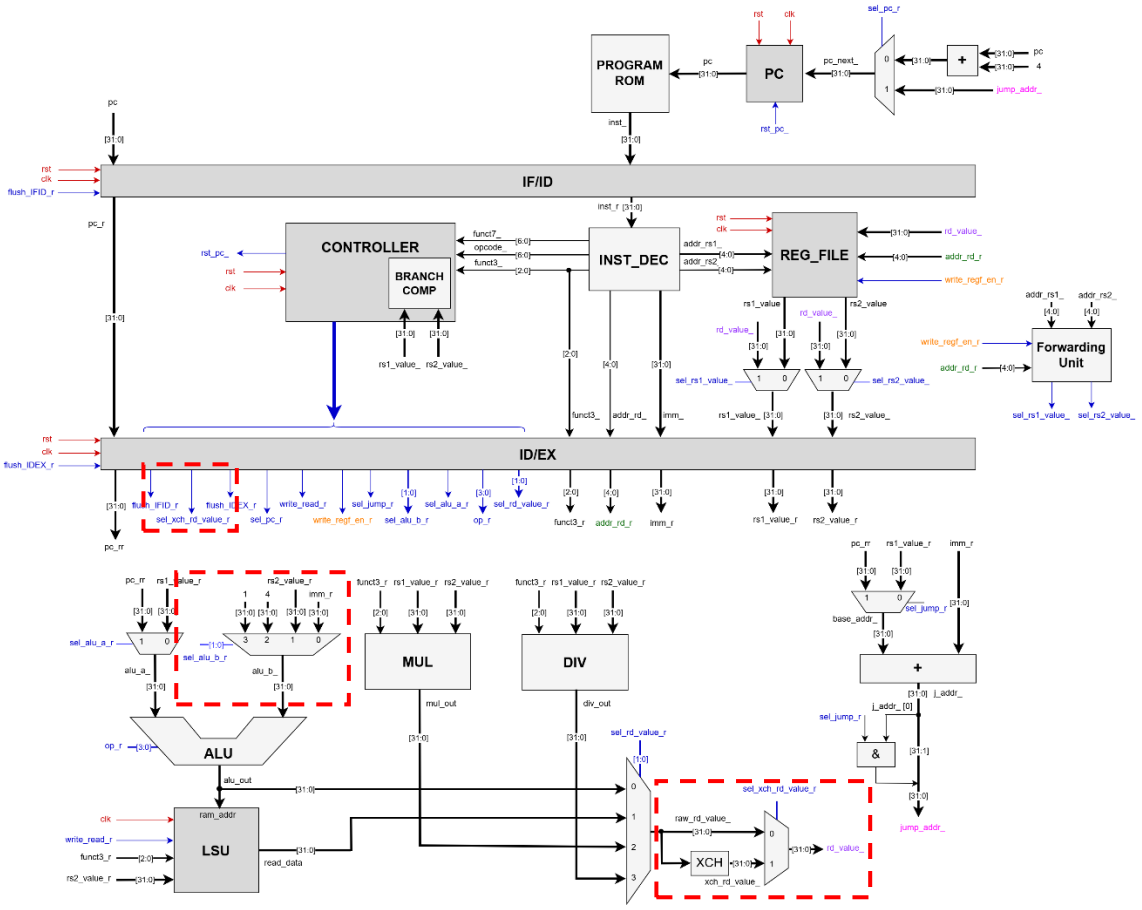
2. 指令說明 :

```
if ( x[rs1] != 0 )
        PC ← PC + imm
        x[rd] ← x[rs1] – 1
else
        PC ← PC + 4
```

CNZDECJ (Check Not Zero then Decrement and Jump) 若 rs1 暫存器內的數值不為零，則跳躍至 PC + sext(imm) 的指令位址並且將 rs1 暫存器的數值減 1 存入 rd 暫存器；否則繼續執行下一個指令 (PC + 4)。

※ 此指令的立即值解碼方式為直接進行**符號擴展**。

架構圖:



testbench:

```
`timescale 1ns/100ps
module testbench;

    logic               clk;
    logic               rst;

    RISCV_Core u_RISCV_Core(
        .clk        (clk    ),
        .rst        (rst    ),
        //
        .regs_31    (       )
    );

    always #5 clk = ~clk;

    initial begin

        clk = 0;
        rst = 1;

        #20 rst = 0;
        #5000 $stop;
    end
endmodule
```

模擬 ：XCH16.R 與 CNZDECJ 指令模擬

Program_ROM :

```
module Program_Rom(
    input   logic [31:0] Rom_addr,
    output  logic [31:0] Rom_data
);

    always_comb begin
        case (Rom_addr)
            32'h0  : Rom_data = 32'h00020137;   //init: LUI      x2, 32
            32'h4  : Rom_data = 32'h00110113;   //       ADDI     x2, x2, 1
            32'h8  : Rom_data = 32'h00202023;   //       SW       x2, 0(x0)
            32'hC  : Rom_data = 32'h00040137;   //       LUI      x2, 64
            32'h10 : Rom_data = 32'h00310113;   //       ADDI     x2, x2, 3
            32'h14 : Rom_data = 32'h00202223;   //       SW       x2, 4(x0)
            32'h18 : Rom_data = 32'h00060137;   //       LUI      x2, 96
            32'h1C : Rom_data = 32'h00510113;   //       ADDI     x2, x2, 5
            32'h20 : Rom_data = 32'h00202423;   //       SW       x2, 8(x0)
            32'h24 : Rom_data = 32'h00200113;   //       LI       x2, 2
            32'h28 : Rom_data = 32'h00000193;   //       ADDI     x3, x0, 0
            32'h2C : Rom_data = 32'h0001a203;   //loop:  LW       x4, 0(x3)
            32'h30 : Rom_data = 32'h000202db;   //       XCH16.R  x5, x4
            32'h34 : Rom_data = 32'h00418193;   //       ADDI     x3, x3, 4
            32'h38 : Rom_data = 32'he801217b;   //       CNZDECJ  x2,x2, loop
            default: Rom_data = 32'h00000013;   //NOP
        endcase
    end
endmodule
```

輸出結果波形圖: