0416315 王定偉、0416005 張彧豪

**Computer Organization Lab3**

**Architecture diagram:**

**Detailed description of the implementation:**

1. 以Simple\_Single\_CPU為主要的模組，會去呼叫這個simple CPU所需要的所有模組，包含PC、IM、RF、Decoder、ALU\_Ctrl、Sign\_Extend、MUX，ALU、Data\_memory等模組。此CPU為一個clock cycle執行一個指令。
2. Decoder模組：
3. 我是根據傳入的 Op filed去map那些輸出的值，以下是我採用的對應表：

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| instr\_op\_i | ALU\_op\_o | Type | MemToReg\_o  MemRead\_o  MemWrite\_o | RegDst\_o  RegWrite\_o | ALUSrc\_o | Branch\_o  Brach\_type\_o  Jump\_o |
| 6’d0 | 3’b000 | R | 2’b00  1’b0  1’b0 | 2’b01  1’b1 | 1’b0 | 1’b0  2’b00  1’b0 |
| 6’d2 | 3’b111 | J | 2’b00  1’b0  1’b0 | 2’b00  1’b0 | 1’b0 | 1’b0  2’b00  1’b1 |
| 6’d3 | 3’b111 | JAL | 2’b11  1’b0  1’b0 | 2’b10  1’b0 | 1’b0 | 1’b0  2’b00  1’b1 |
| 6’d4 | 3’b001 | BEQ | 2’b00  1’b0  1’b0 | 2’b00  1’b0 | 1’b0 | 1’b1  2’b00  1’b0 |
| 6’d5 | 3’b010 | BNE,  BNEZ | 2’b00  1’b0  1’b0 | 2’b00  1’b0 | 1’b0 | 1’b1  2’b11  1’b0 |
| 6’d6 | 3’b001 | BLT | 2’b00  1’b0  1’b0 | 2’b00  1’b0 | 1’b0 | 1’b1  2’b10  1’b0 |
| 6’d7 | 3’b001 | BLE | 2’b00  1’b0  1’b0 | 2’b00  1’b0 | 1’b0 | 1’b1  2’b01  1’b0 |
| 6’d8 | 3’b011 | ADDI | 2’b00  1’b0  1’b0 | 2’b00  1’b1 | 1’b1 | 1’b0  2’b00  1’b0 |
| 6’d13 | 3’b100 | ORI | 2’b00  1’b0  1’b0 | 2’b00  1’b1 | 1’b1 | 1’b0  2’b00  1’b0 |
| 6’d15 | 3’b101 | LI | 2’b10  1’b0  1’b0 | 2’b00  1’b1 | 1’b1 | 1’b0  2’b00  1’b0 |
| 6’d35 | 3’b011 | LW | 2’b01  1’b1  1’b0 | 2’b00  1’b1 | 1’b1 | 1’b0  2’b00  1’b0 |
| 6’d43 | 3’b011 | SW | 2’b00  1’b0  1’b1 | 2’b01  1’b0 | 1’b1 | 1’b0  2’b00  1’b0 |

我們的ALU\_op\_o與instr\_op\_i之間並沒有什麼特殊的關係，純粹只是循序的指定。

1. ALU\_Control模組：
2. 這個ALU\_Control模組跟Lab2得非常相似，差異在於這個ALU\_Control有多一條輸出線為Junp\_type，這條輸出線的用意是要幫助我們實作的JR指令用的。
3. 以下為我們採用的對應表：

R-type 以外：

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ALUOp\_i | Instruction | ALUSrc\_1\_o | ALUCtrl\_o | Jump\_type |
| 3’b001 | BEQ,BLT,BLE | 1’b0 | 4’b0110 | 1’b0 |
| 3’b010 | BNE,BNEZ | 1’b0 | 4’b1110 | 1’b0 |
| 3’b011 | Addi,LW,SW | 1’b0 | 4’b0010 | 1’b0 |
| 3’b100 | LI | 1’b0 | 4’b0100 | 1’b0 |
| 3’b101 | ORI | 1’b0 | 4’b0001 | 1’b0 |

這五個指令除了BEQ、BLT、BLE、BNE、BNEZ、Addi、LW、SW的ALUCtrl\_o是拿lab1中減法、加法做對應，其它是拿尚未被佔用的指令去指派的。

R-type：

|  |  |  |  |
| --- | --- | --- | --- |
| ALUOp\_i | Instruction | ALUSrc\_1\_o | ALUCtrl\_o |
| 3’b000 | ADD | 1’b0 | 4’b0010 |
| SUB | 1’b0 | 4’b0110 |
| AND | 1’b0 | 4’b0000 |
| OR | 1’b0 | 4’b0001 |
| SLT | 1’b0 | 4’b0111 |
| SLTU | 1’b0 | 4’b1111 |
| MUL | 1’b0 | 4’b0101 |
| JR | 1’b0 | 4’b1000 |
| SLL | 1’b1 | 4’b0011 |
| SLLV | 1’b0 | 4’b0011 |

R-type中前五個指令是照lab1去指定的，而後面三個指令是拿尚未被佔用的值去指定的。

1. Branch mux詳細說明：
2. JAL實作方法：

Lal

1. JR實作方法：

lal

**Problems encountered and solutions:**

**Lesson learnt (if any):**