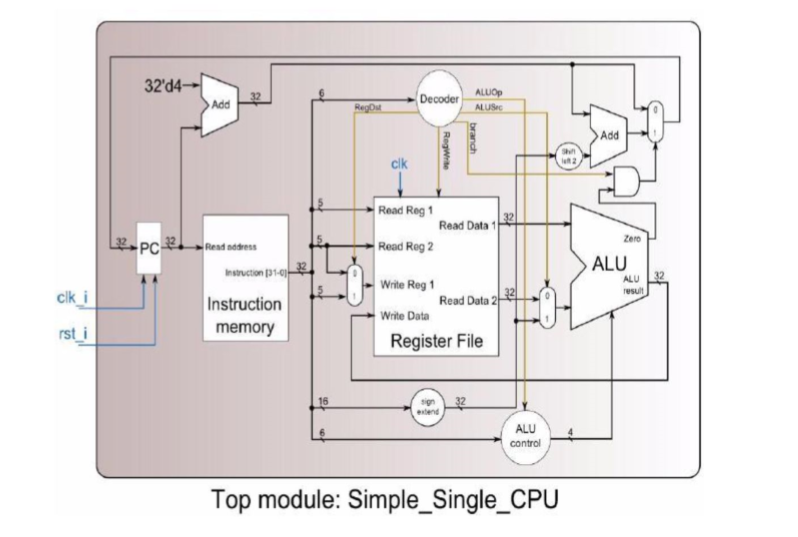
**Computer Organization**

**Architecture diagrams:**



**Hardware module analysis:**

**Program Counter：**

**目前指令的位置。**

**Decoder：**

**判斷目前所要執行的operation為何，輸出RegWrite\_o/** **ALU\_op\_o/** **ALUSrc\_o/** **RegDst\_o/** **Branch\_o等控制訊號。**

**Instruction Memory：**

**將目前要執行的指令存到Instruction Memory。**

**Register File：**

**依照R-type/I-type來讀取1-2個register檔案內容並輸出做運算。**

**ALU：**

**依據收到ALU Control的訊號，將輸入的值作相對應的operation。**

**Adder：**

**Adder1**

**將pc的指向下一個指令(pc+4)**

**Adder2**

**依據分支給予的偏移量加上原本pc的值來更新pc的值。**

**ALU Control：**

**控制ALU所要做的operation。**

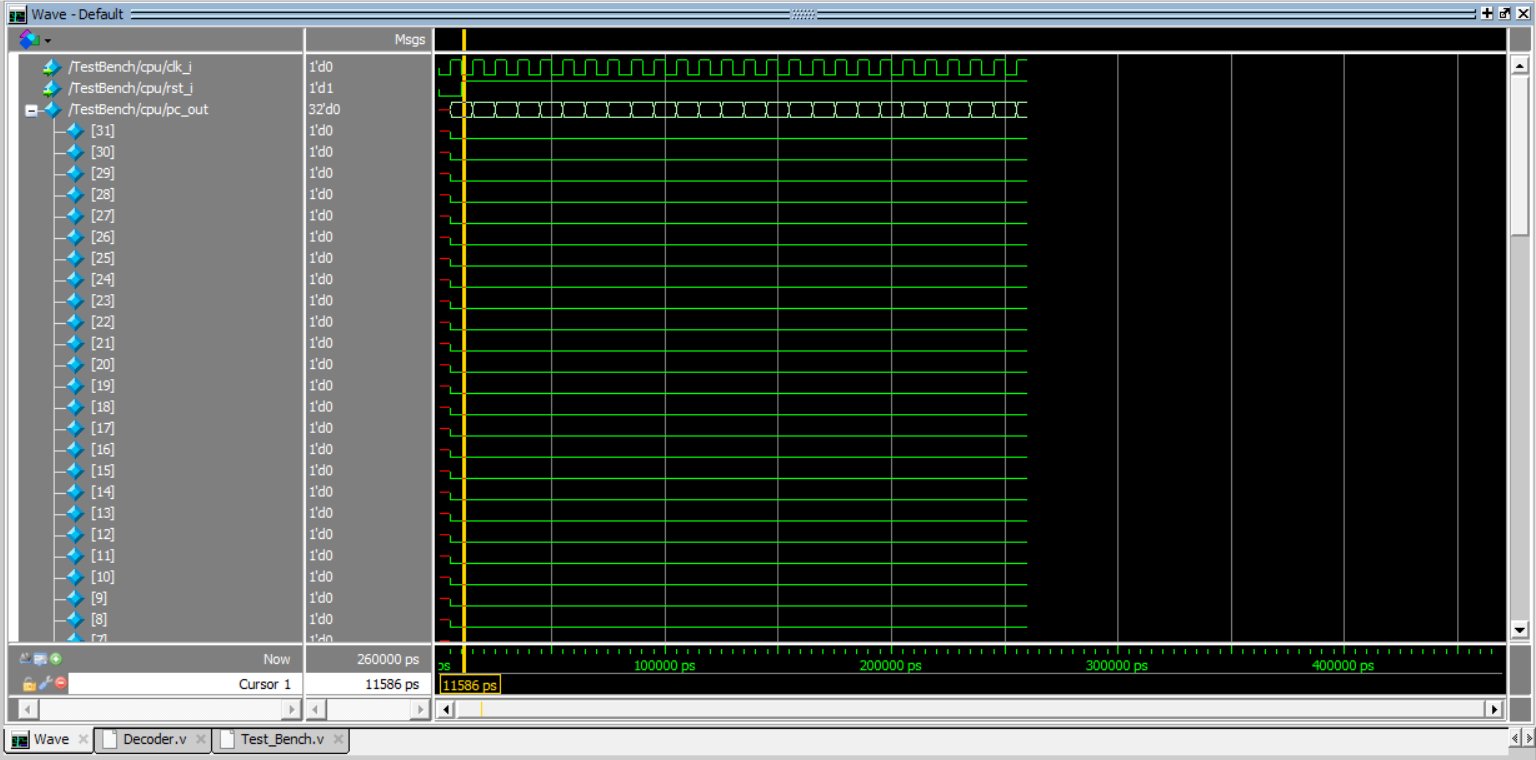
**Shift Left 2：**

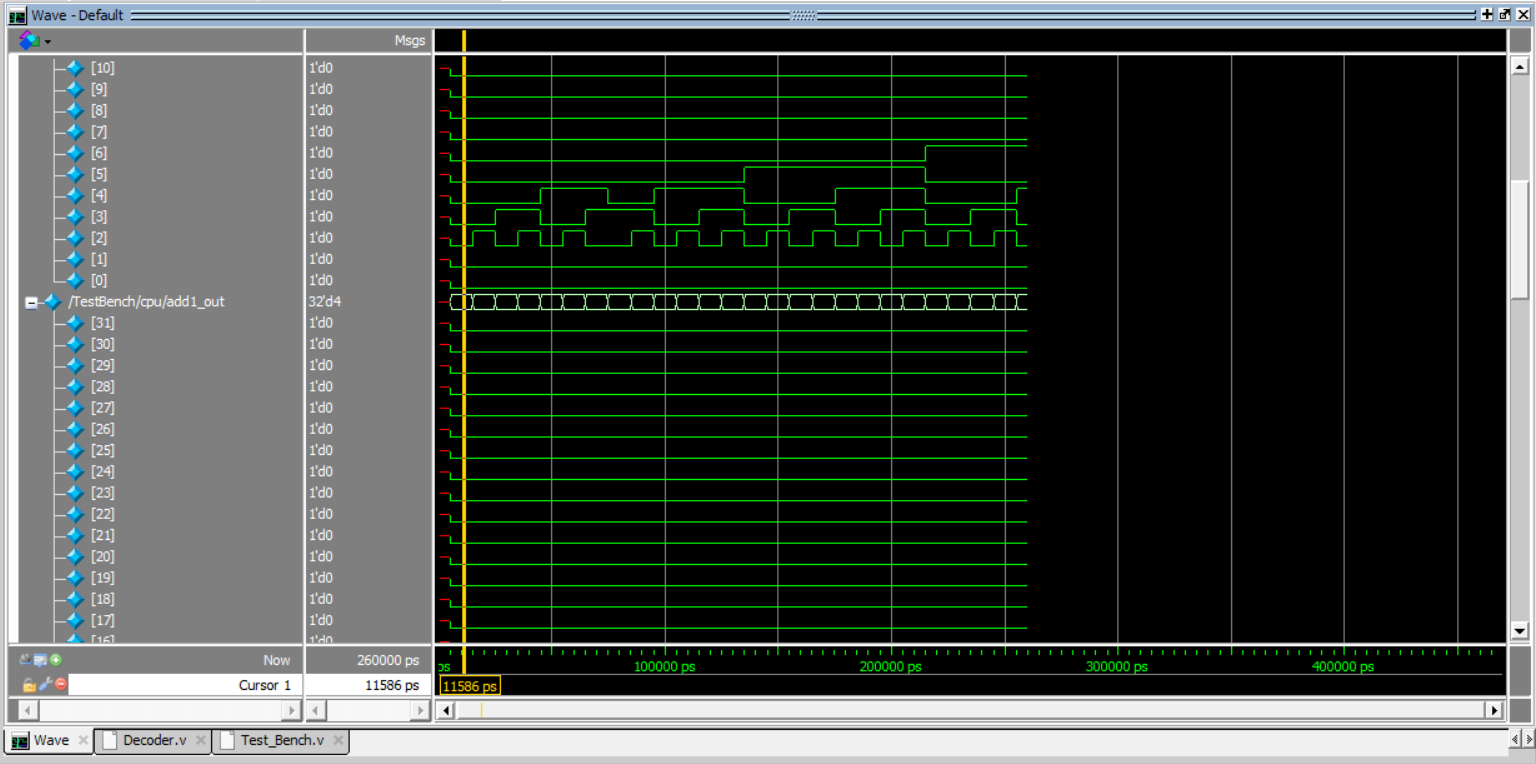
**因為instruction的單位為word(32 bit)，所以要向左移2 bit將單位轉為以byte(8 bit)來計算**

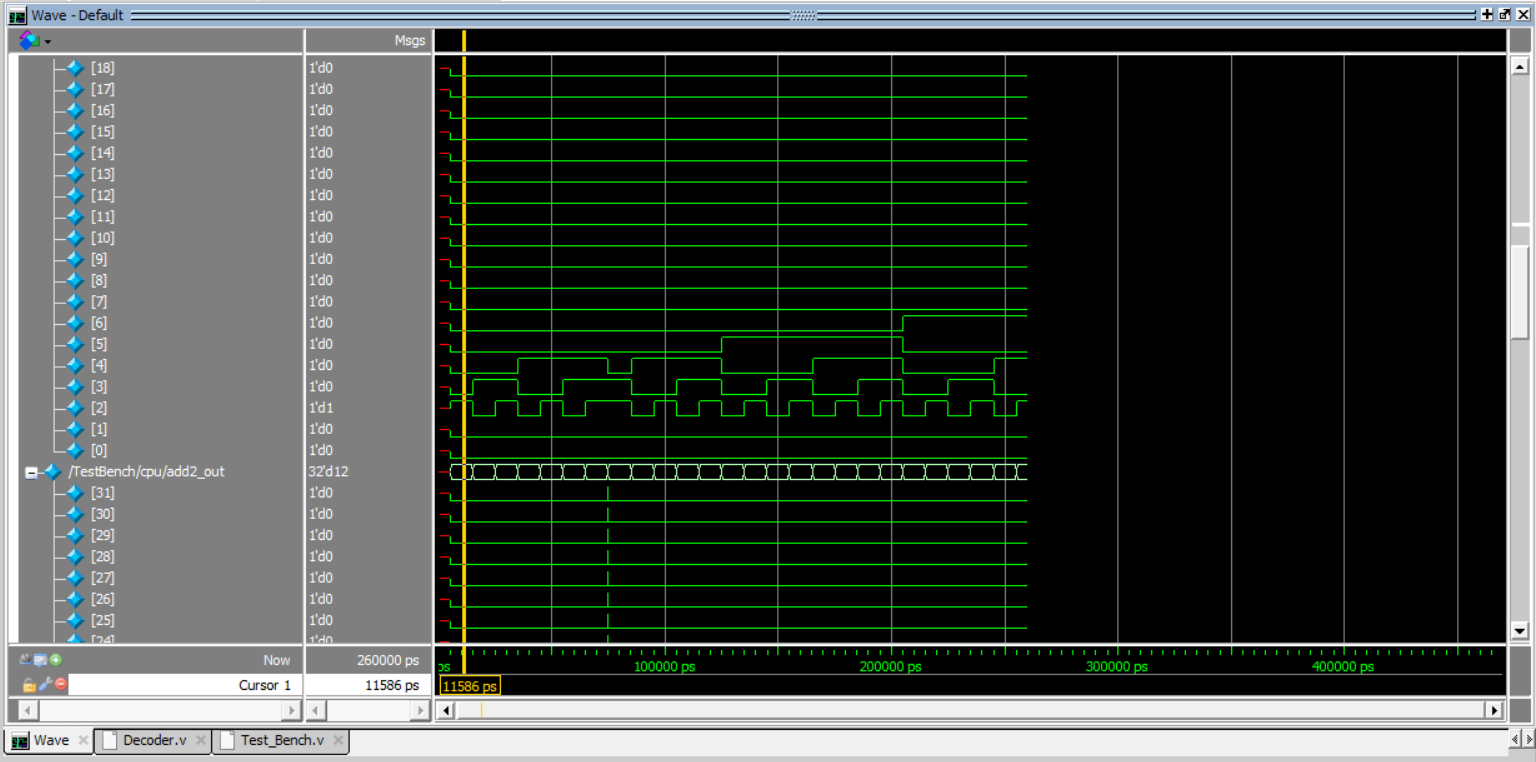
**Sign Extend：**

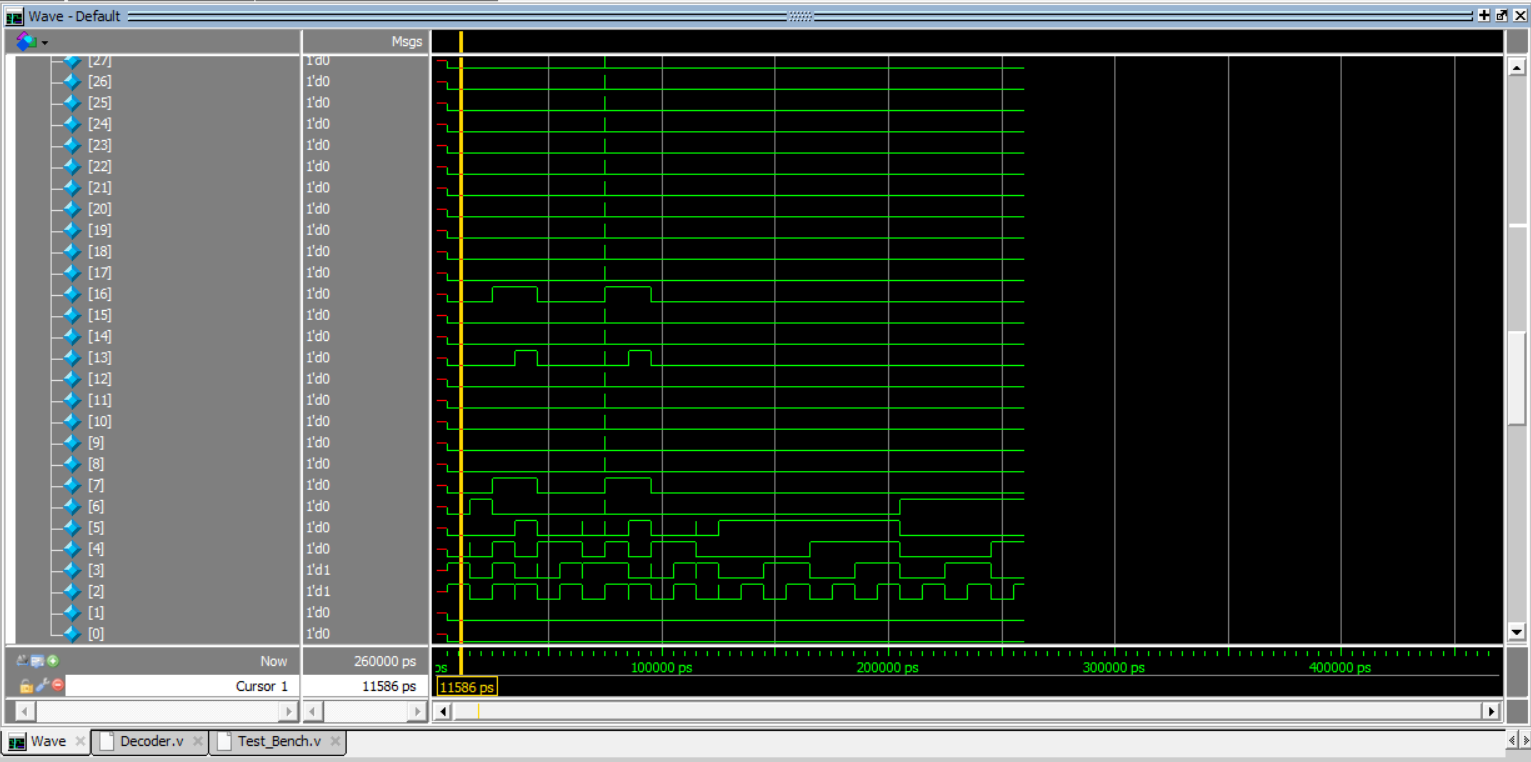
**像是immediate等的值在Iinstruction中只存放16 bit，因此需要透過sign extend為32 bit才能當作輸入。**

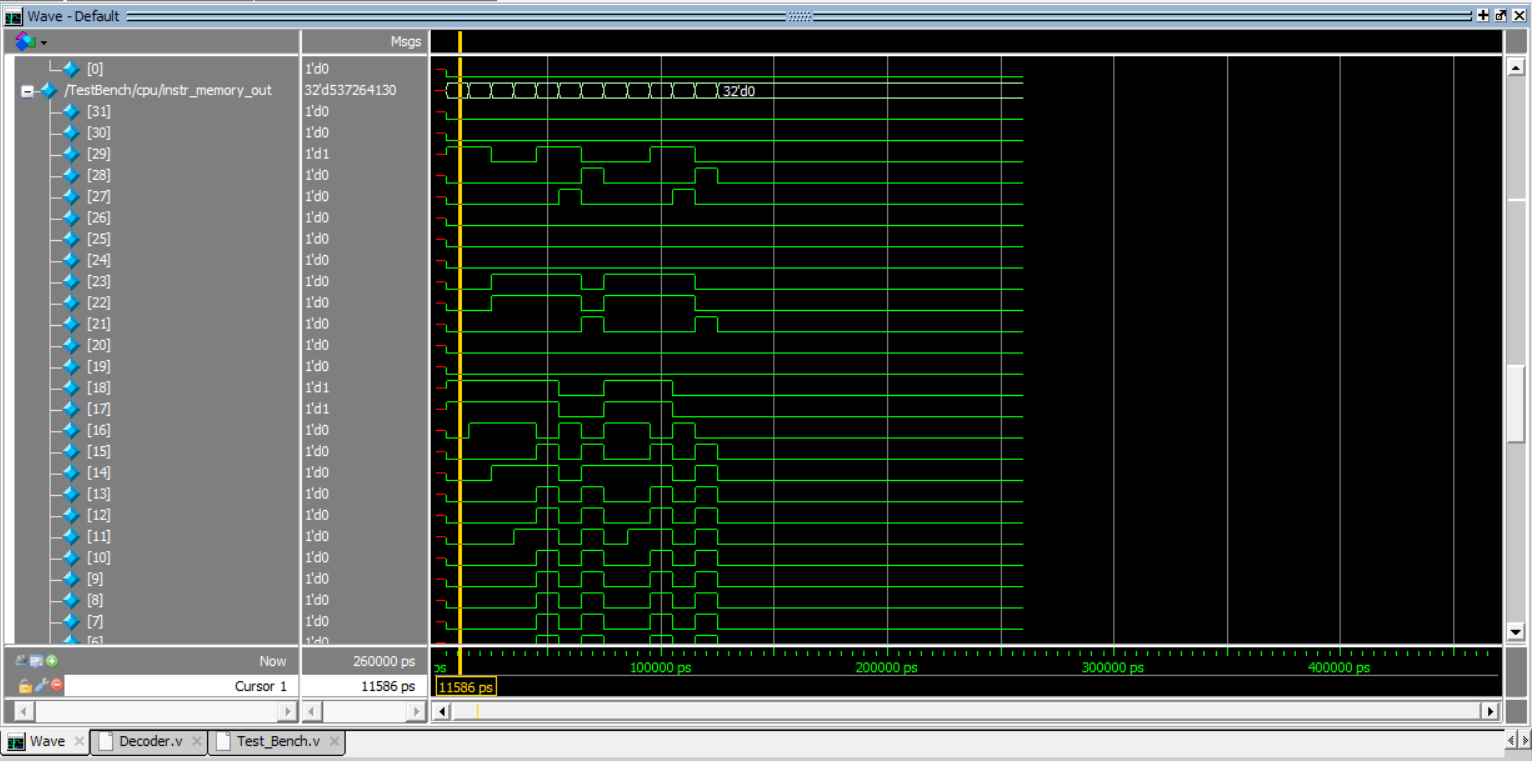
**Finished part:**

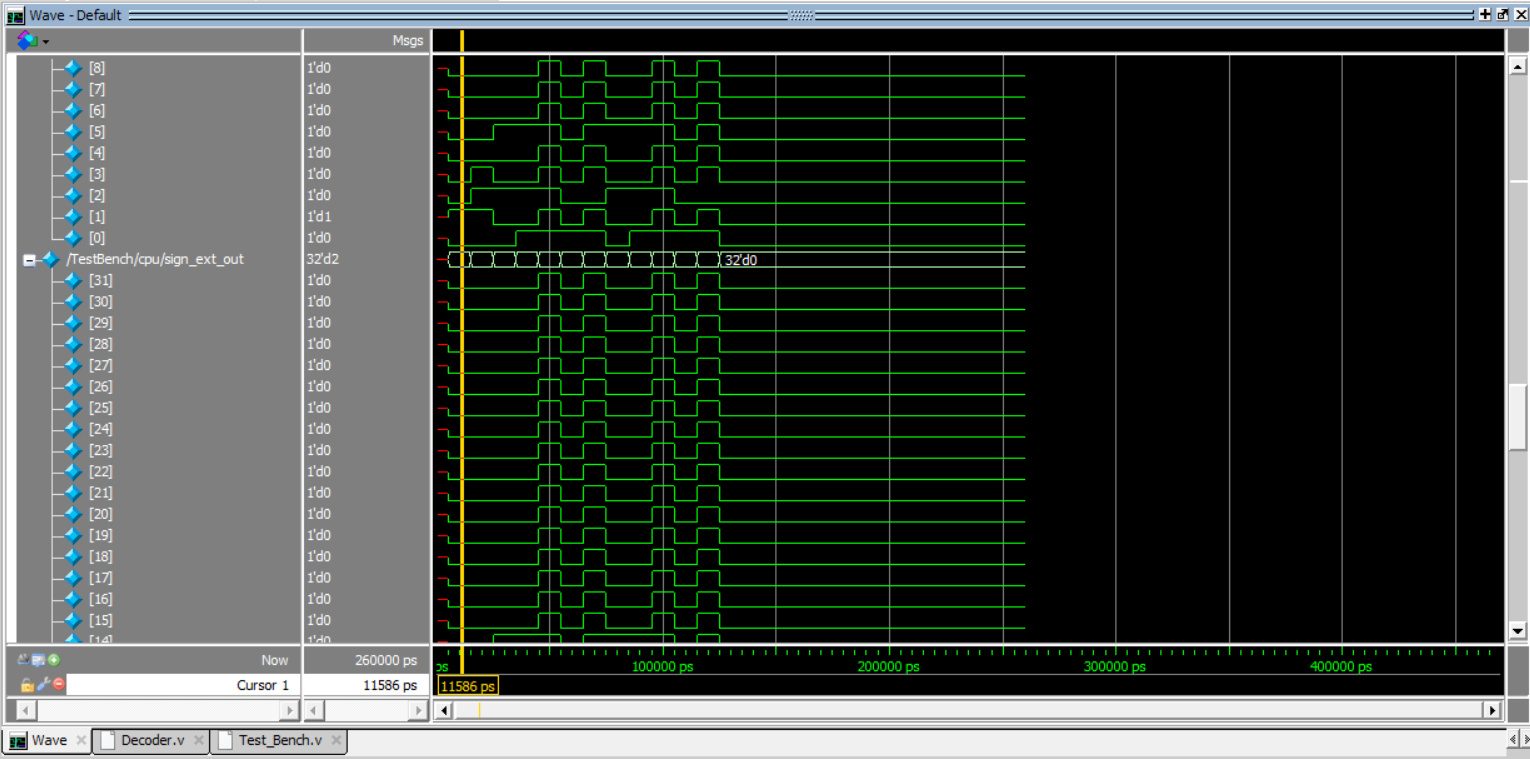


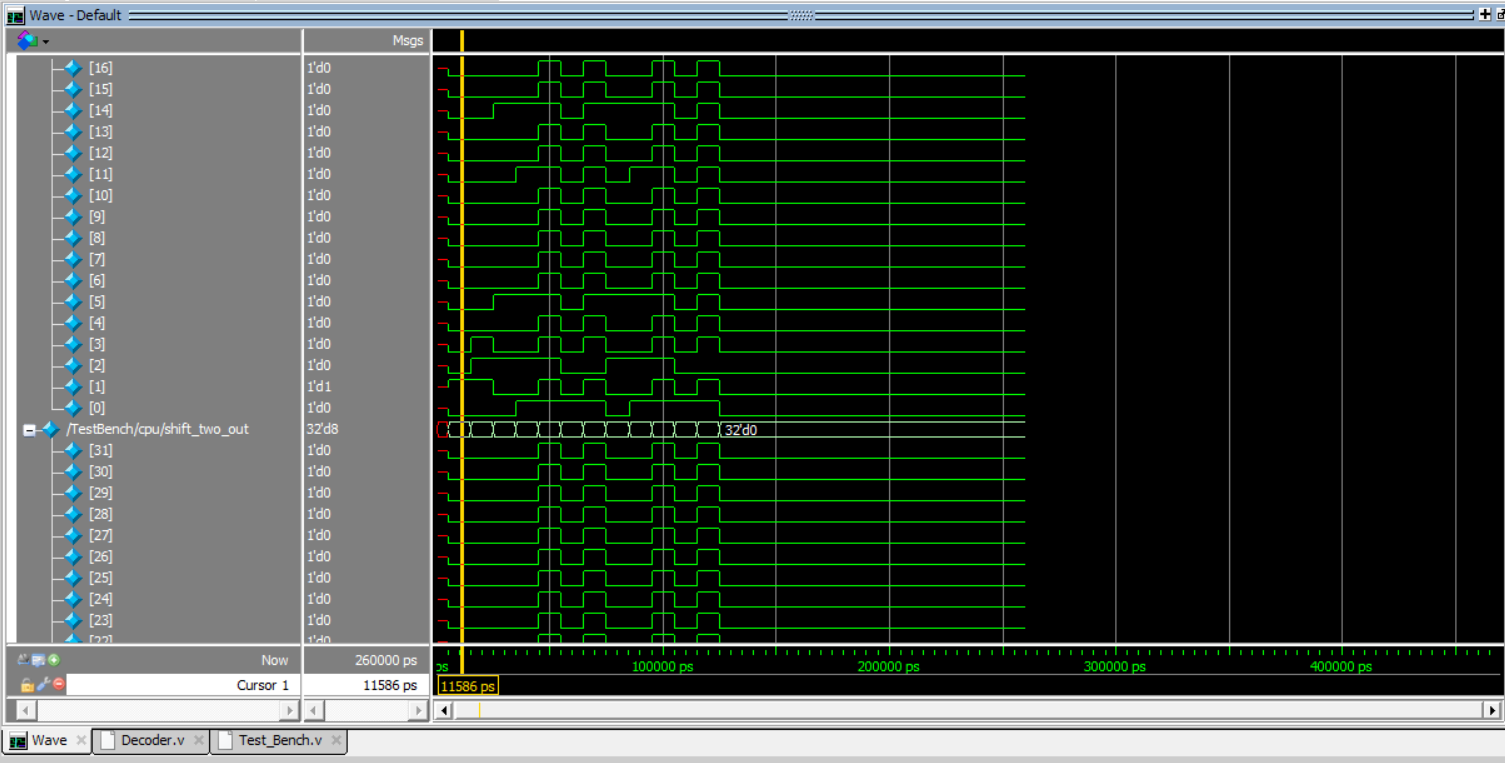


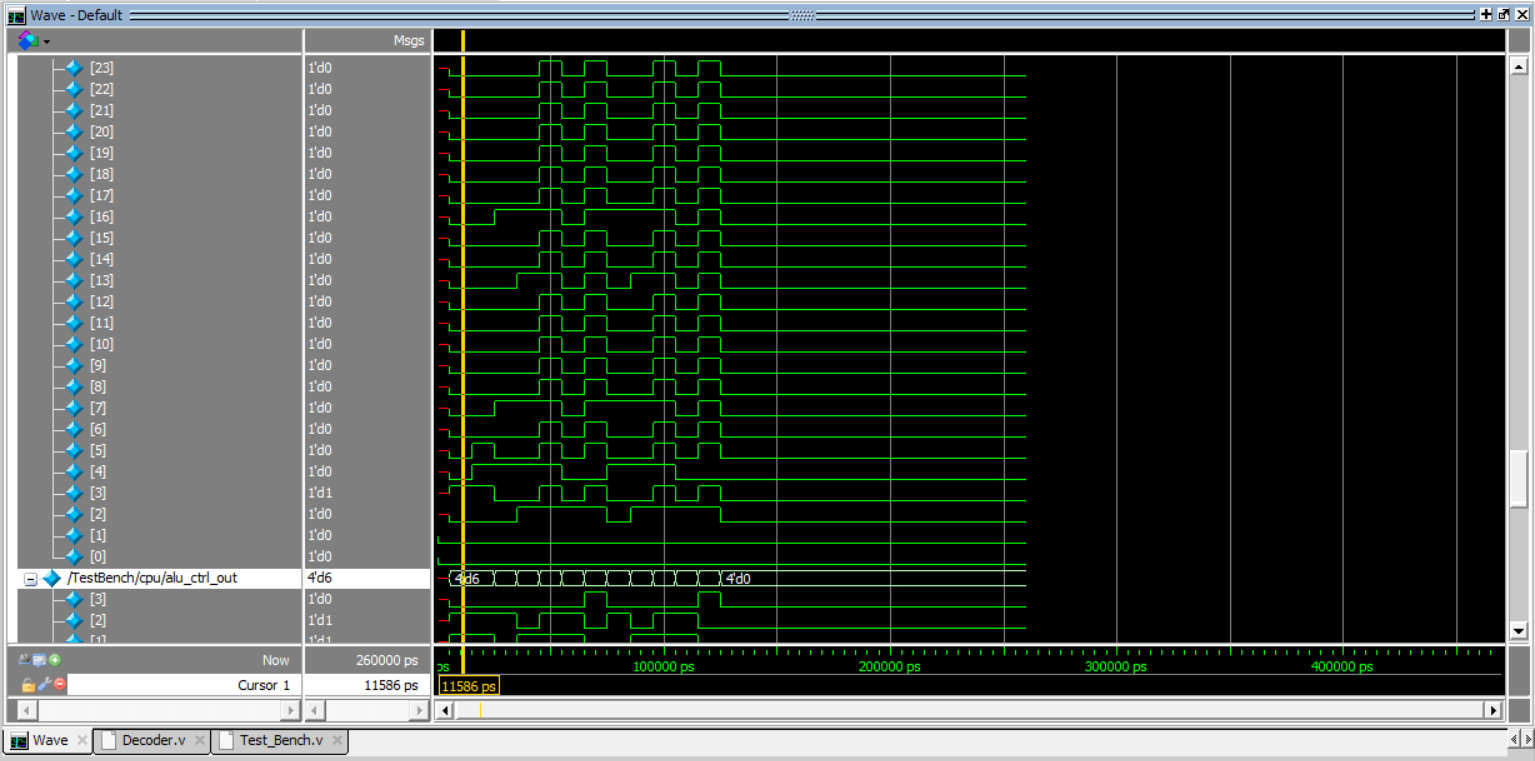


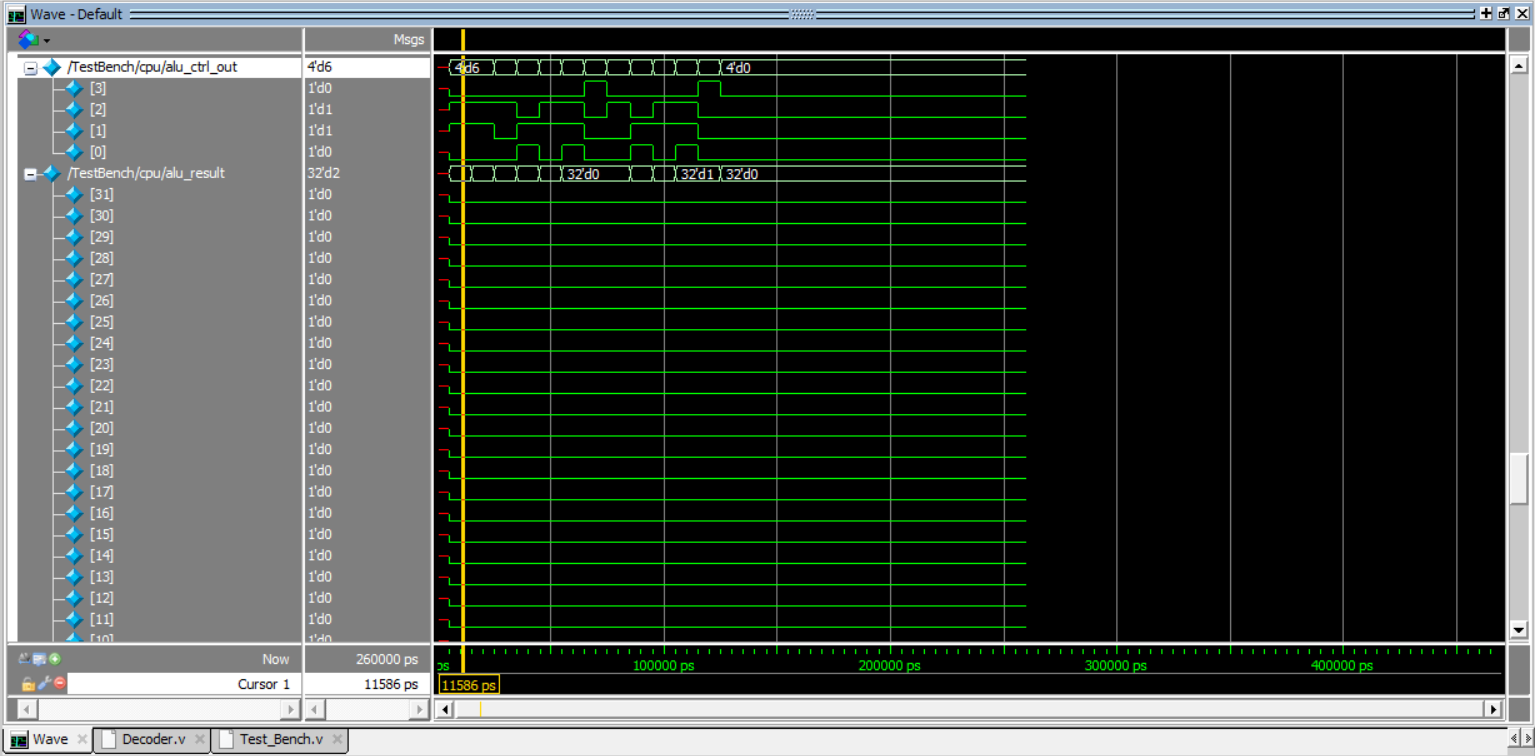


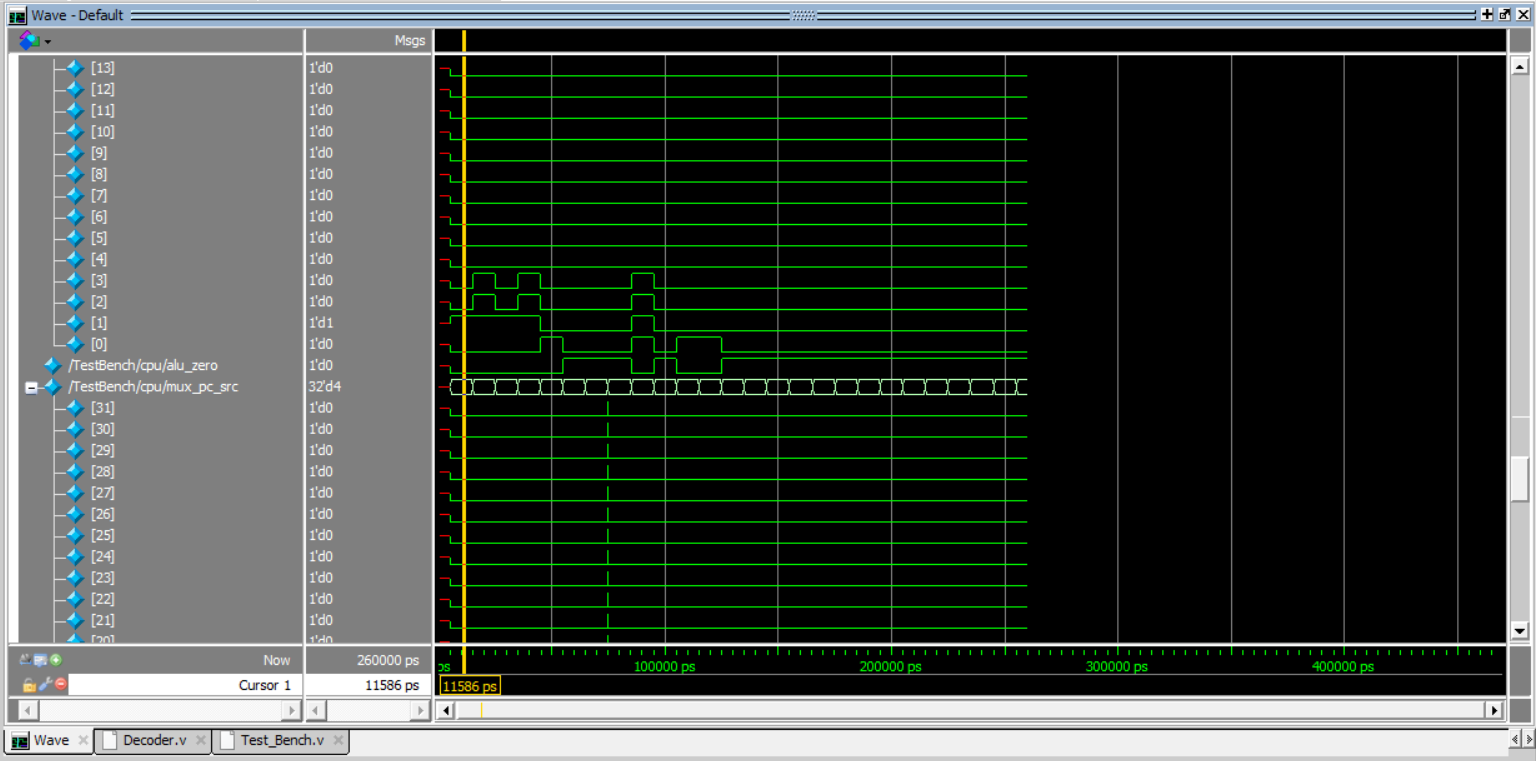


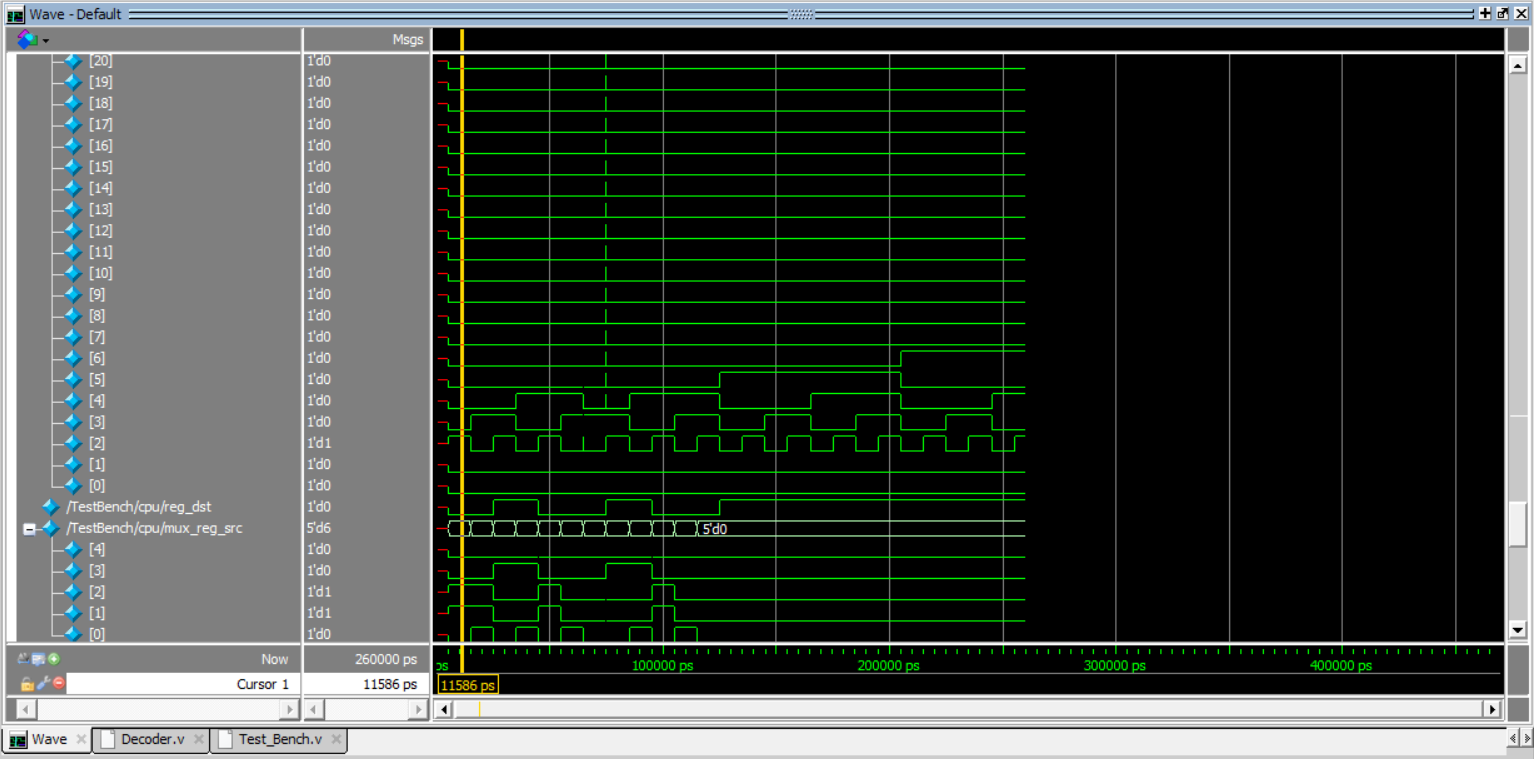


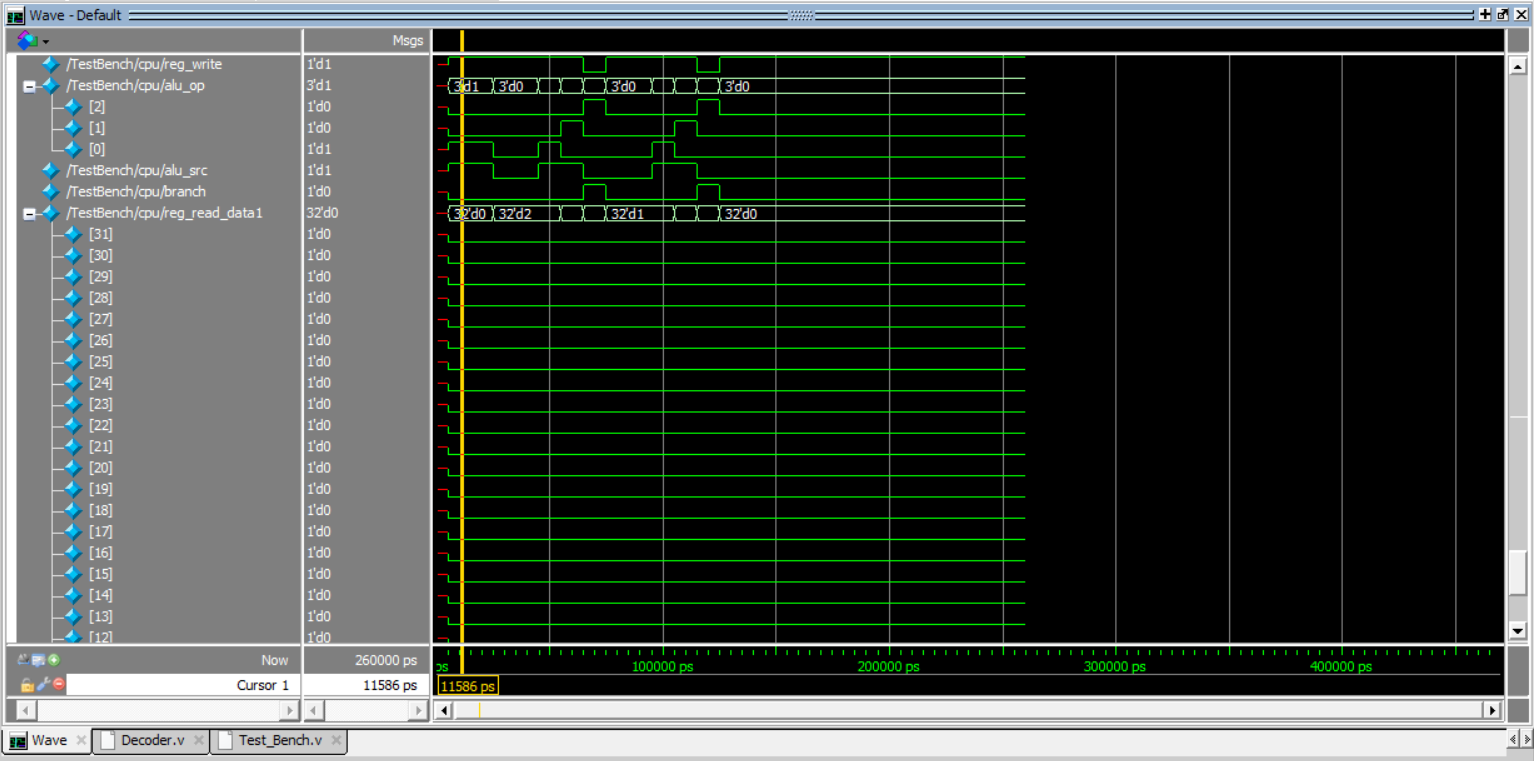


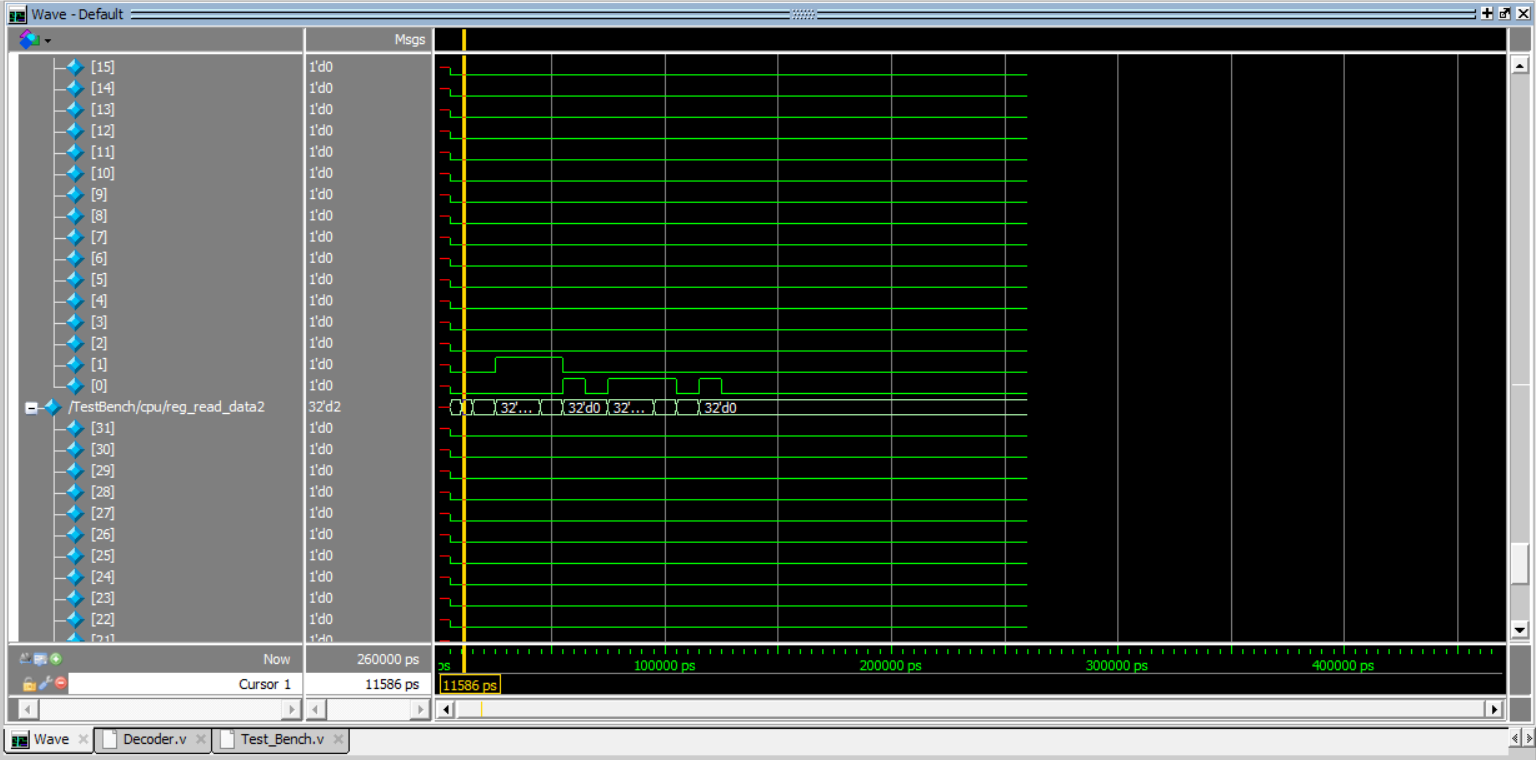


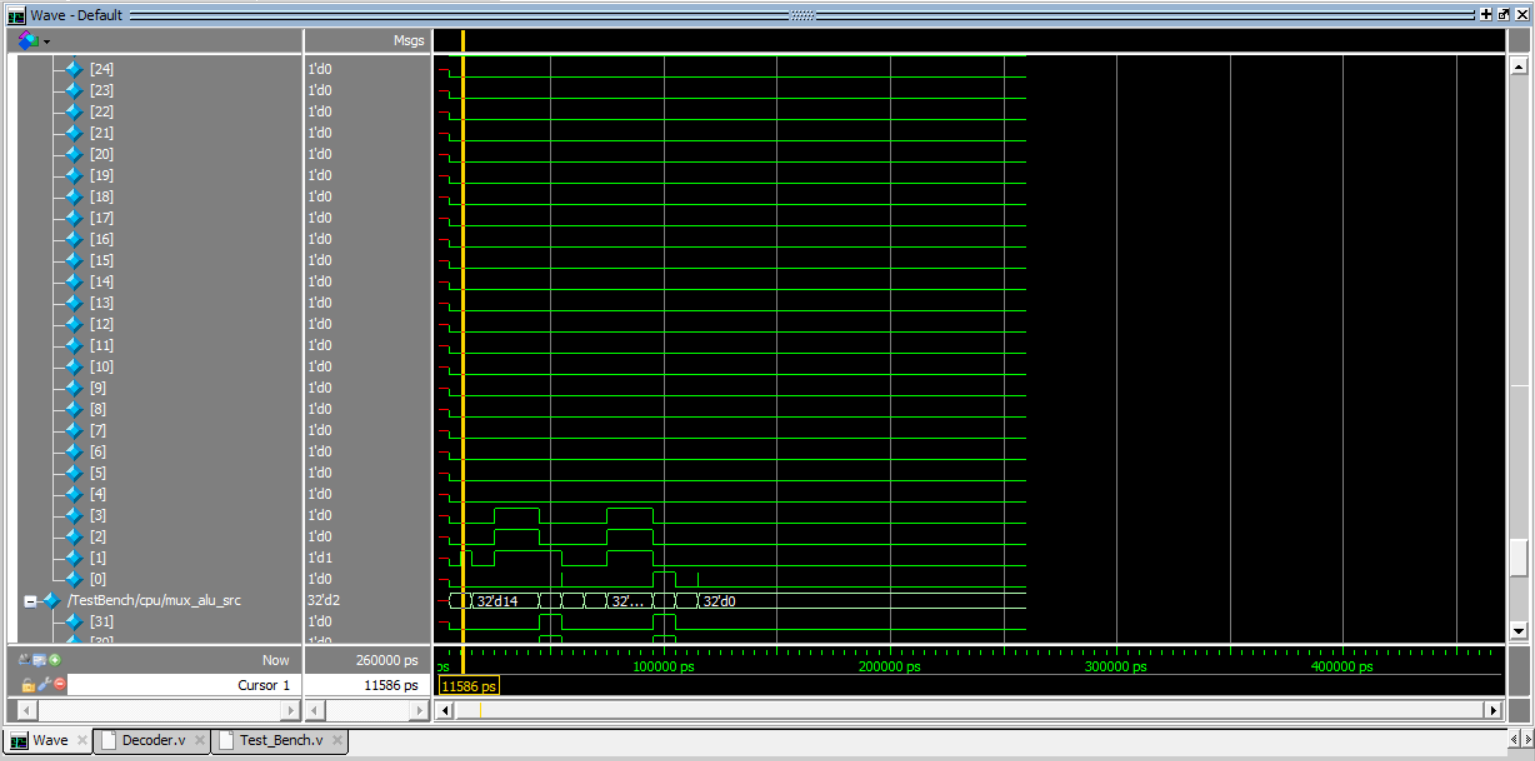


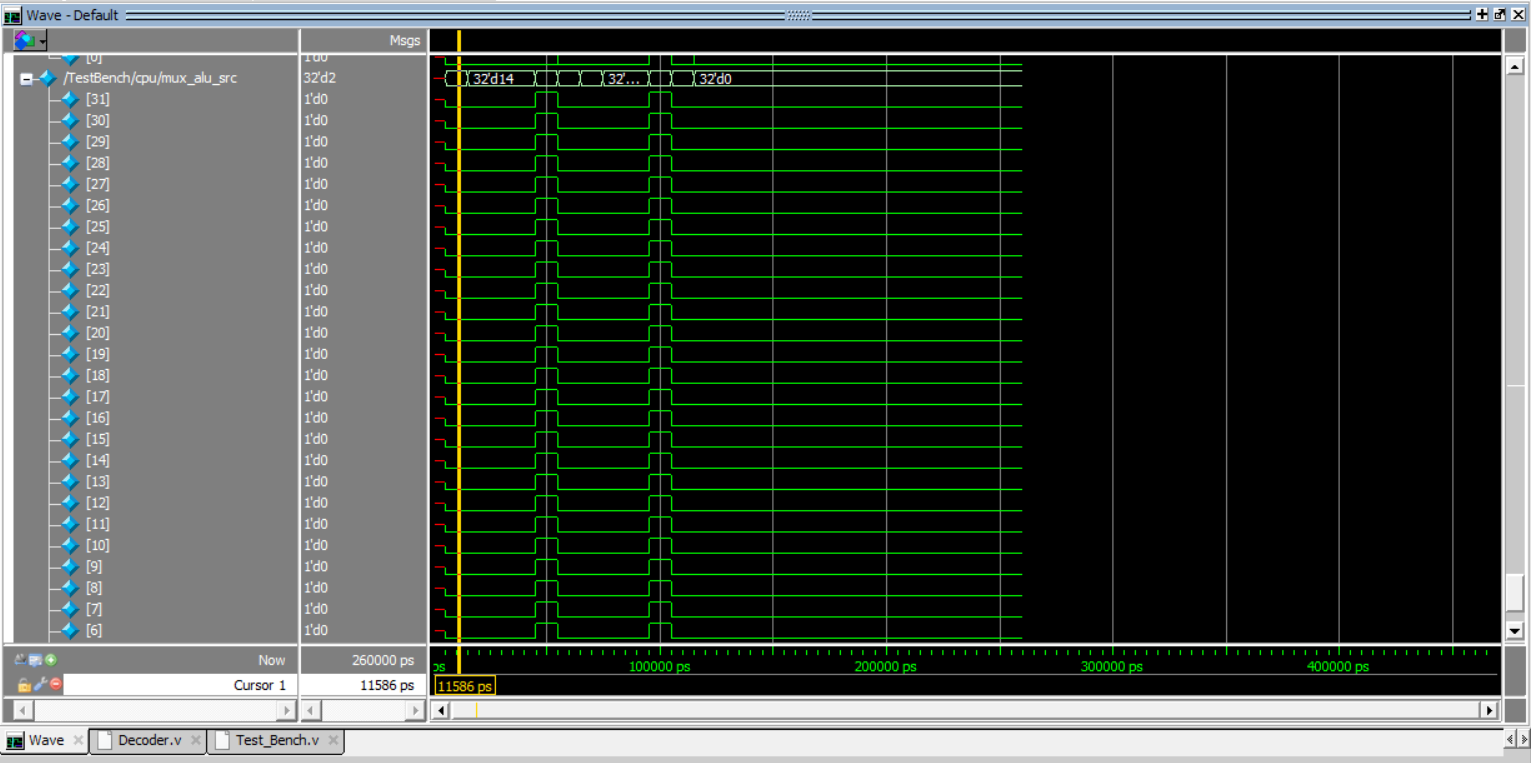


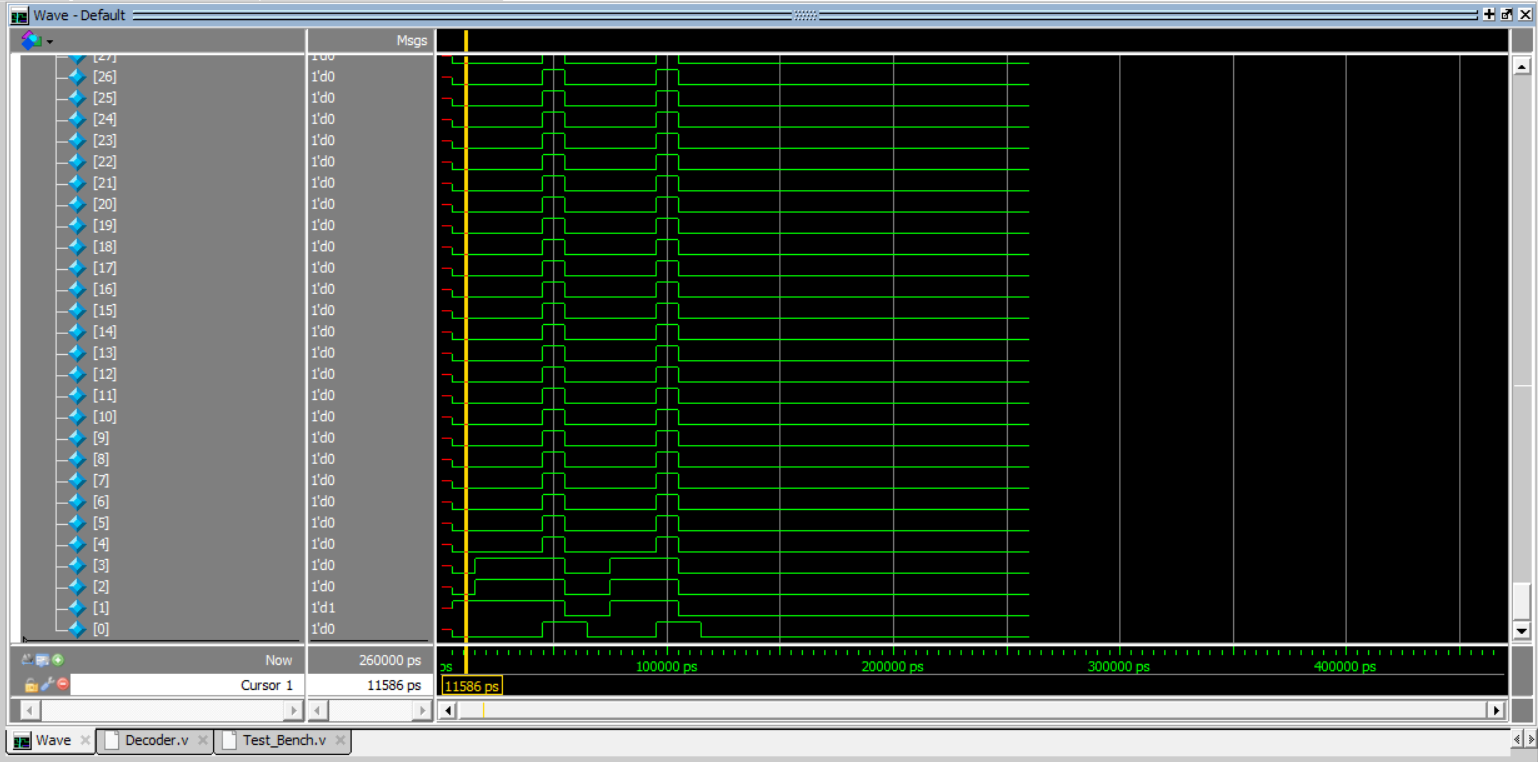












**Problems you met and solutions:**

1. **在印出result\_o結果時，會將r0的值改掉，並且即便已經執行完instruction後，在波形圖中result\_o的值仍會變動**

**Sol：在alu.v判斷要執行哪個operation的最後加上default : result\_o <= 1'b0;**

1. **在印出波型圖時，發現不同instruction其對應的某部分布林值相反**

**Sol：1.將beq若相等則設的result\_o判斷式寫相反了**

**2.將MUX\_2to1內的判斷式寫相反了**

1. **透過波型圖在判斷operation所應執行的動作時，發現有些op code明明是對的卻不會執行正確的operation**

**Sol：在decoder中的output值的判斷式沒有將相對應的instr\_op\_i考慮周全，因此輸出ALU\_op\_o時有誤。**

**Summary:**

**這次lab花了比較多時間在debug上面，主要都是錯在邏輯式的判斷上。在寫每個module各自的內容都滿好寫的，不過在寫到Simple\_Single\_CPU才發現自己對於instruction的control unit都很模糊，因此在將整個CPU架構起來的同時有點不知道該如何下手，這次實地寫了lab之後才弄清楚mux與instruction每個部份的操作。**