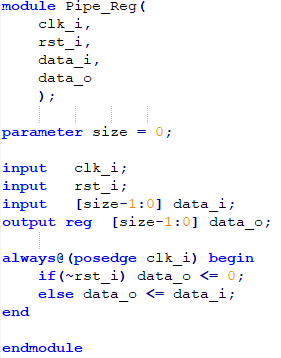
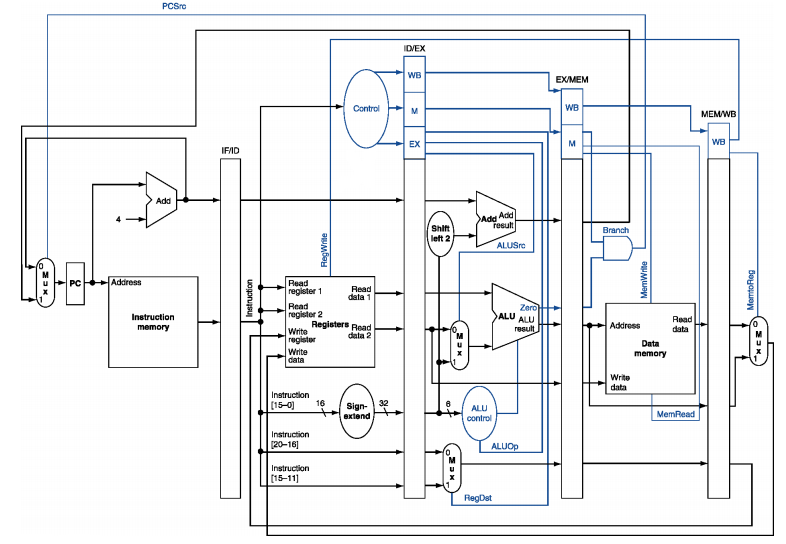
**Computer Organization Lab4**

**Source code and the note:**

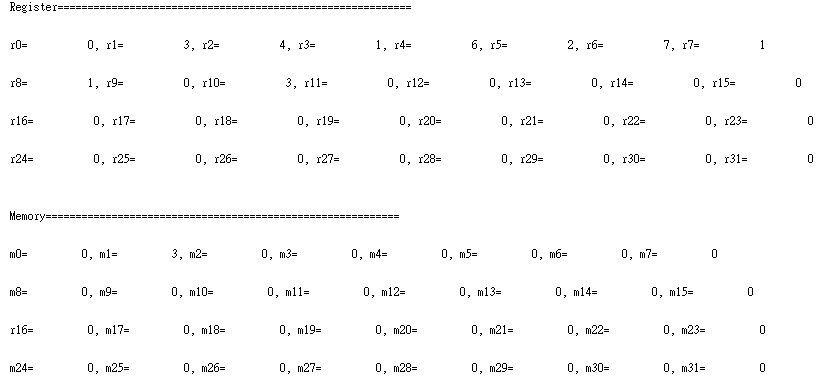
這次作業中我認為最重要的module就是Pipe\_Reg，藉由這個module我們才得以實現Pipeline CPU，在其中可以注意到我們使用<=而並非=是因為我們需要我們的訊號能夠在register中維持一個clock cycle的時間，而這是<=才能做到的，若使用=的話，data\_i一變data\_o就會馬上改變，也不會使我們希望的結果。

**Architecture diagrams:**



**Hardware module analysis:**

這次的電路圖與Lab2的大致相同，最大的差別是為了要做出Pipeline CPU，我們在不同stage之中新增了許多register，讓CPU可以一次處理多個指令，同時控制的訊號也是與single cycle一樣正確無誤。

**Finished part:**

**Bonus:**

**Machine Code:**

**00100000000000010000000000010000 addi $1, $0, 16**

**00100000000000110000000000001000 addi $3, $0, 8**

**00000000000000000000000000000000 NOP**

**00000000000000000000000000000000 NOP**

**10101100000000010000000000000100 sw $1, 4($0)**

**00100000001000100000000000000100 addi $2, $1, 4**

**00000000011000010011000000100000 add $6, $3, $1**

**10001100000001000000000000000100 lw $4, 4($0)**

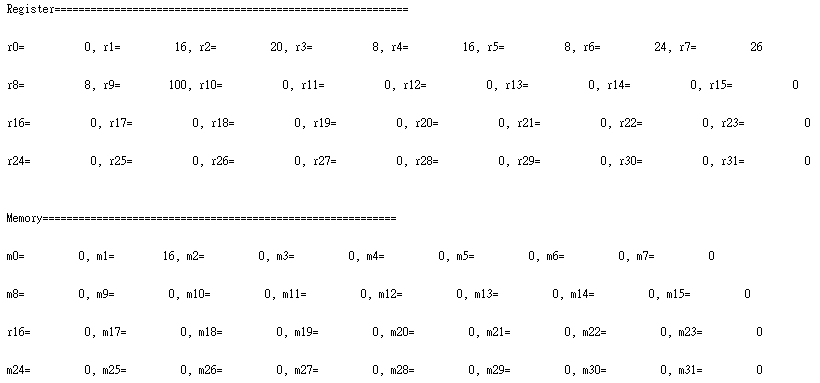
**00100000001001110000000000001010 addi $7, $1, 10**

**00100000000010010000000001100100 addi $9, $0, 100**

**00000000100000110010100000100010 sub $5, $4, $3**

**00000000111000110100000000100100 and $8, $7, $3**

**Finish Part:**

****