**計算機組織LAB03心得:**

10000000 00000000 00000000 00000000 (LOAD M0 R0)

01000000 00000000 00000001 00000000 (STORE M0 R1)

00100000 00000010 00000011 00000001 (ADD R2 R3 R1)

00001000 00000001 00000010 00000000 (SL R1 2 R0)

Test case:

R0 =1; R1=1; R2=2; R3=3;

M0=4; M1=1; M2=2;M3=3:M4=5;M5=5;M6=6;M7=7;

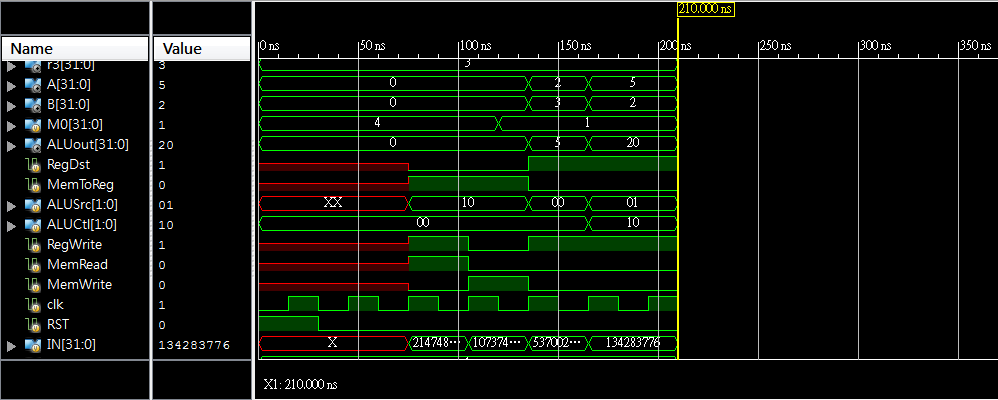
LOAD M0,R0 ; R0 : 1->4

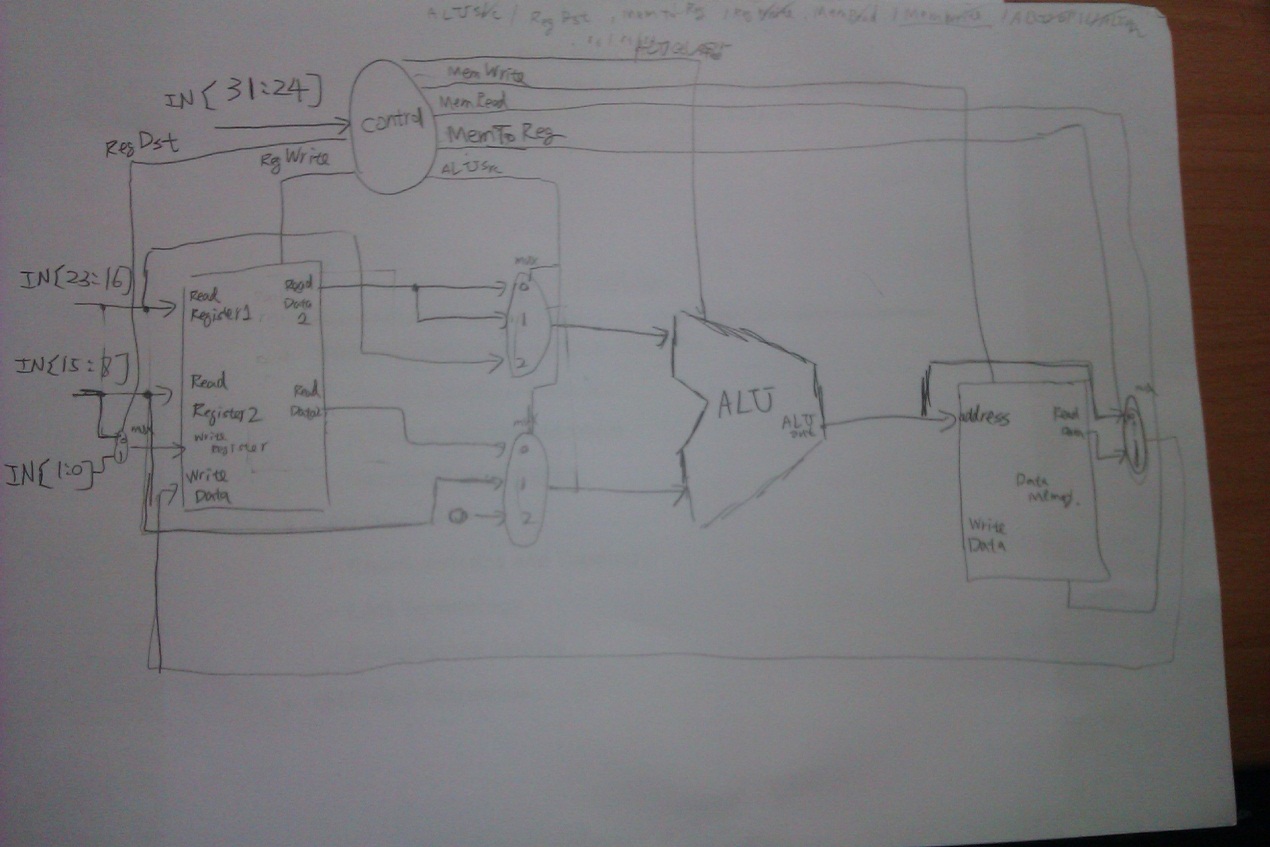
STORE M0,R1; M0:4->1

ADD R2,R3,R1; R1:1->5

SL R1,2,R0 R0:4->20

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **I/O** | **Load** | **Store** | **ADD** | SL | **SL** |
| Input  Op7 1 0 | | | 0 0 | | |
| Op6 0 1 0 0 | | | | | |
| Op5 0 0 1 0 | | | | | |
| Op4 0 0 0 0 | | | | | |
| Op3 0 0 0 1 | | | | | |
| Op2 0 0 0 0 | | | | | |
| Op1 0 0 0 0 | | | | | |
| Op0 0 0 0 0 | | | | | |
| Output | | |  | | |
| ALUSrc 2 2 0 1 | | | | | |
| RegDst 0 0 1 1 | | | | | |
| MemToReg 1 X 0 0 | | | | | |
| RegWrite 1 0 1 1 | | | | | |
| MemRead 1 0 0 0  MemWrite 0 1 0 0  ALUOP1 0 0 0 1  ALUOP0 0 0 0 0  上半 | | | | | |
|  | | | | | |
| 下半 | | | | | |





心得:

先一個一個解釋我產生訊號的用途

ALUSrc:這是用來決定input進ALC的source。因為助教給的format跟課本有點不同，所以我用兩個bit來判斷。

00: A,B都讀進register的值(for add and sub)

01: A讀進register的值，B讀入IN[15:8]當作shieft amount(for shift left and shift right)

10: A讀入IN[23:16]當作memory address，B放入0(因為沒有base register)，這是for load and store。

其實這部分我想了很久，但我仍舊想不出要怎麼用一個bit 實作，因為需要直接讀入常數的是放在不同欄位(memory address和shift amount)，所以就只好用兩個bits。

RegDst: 決定用Rt or Rd field裡寫的register寫入。

MemToReg: 決定要用ALUout寫入register或memory。

RegWrite: 要不要寫入register。

MemRead: 要不要讀memory

MemWrite: 要不要寫入memory

然後是心得:

其實我這次為了弄懂這次作業花了蠻多時間的，可能是還沒這麼熟吧。各個訊號間的傳遞也花了些時間才搞懂。不過寫完之後也有大徹大悟的感覺。也順便去看了課本裡的各個指令的data path。

其實這次作業因為只有四個指令，所以很多signal沒有用到，如: MemToReg，其實不需要用它也可以判斷要怎麼操作，不過為了之後Lab著想我還是有把它寫進它該在的位置。聽助教說之後LAB是single cycle CPU和pipe line，看來我應該好好預習一下，免得又花這麼多時間。