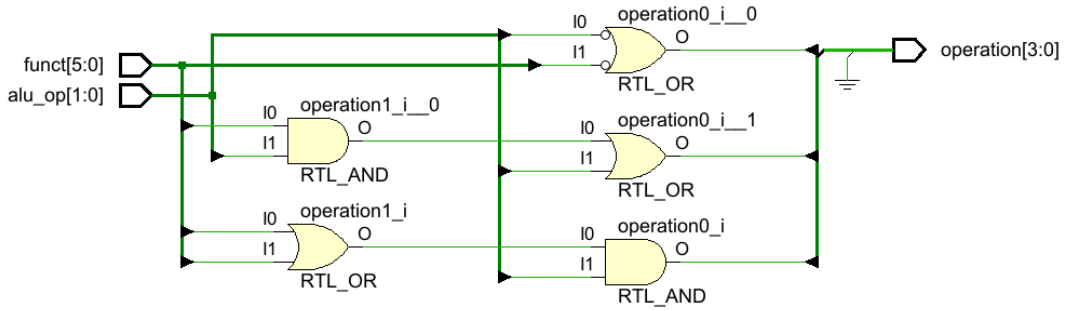
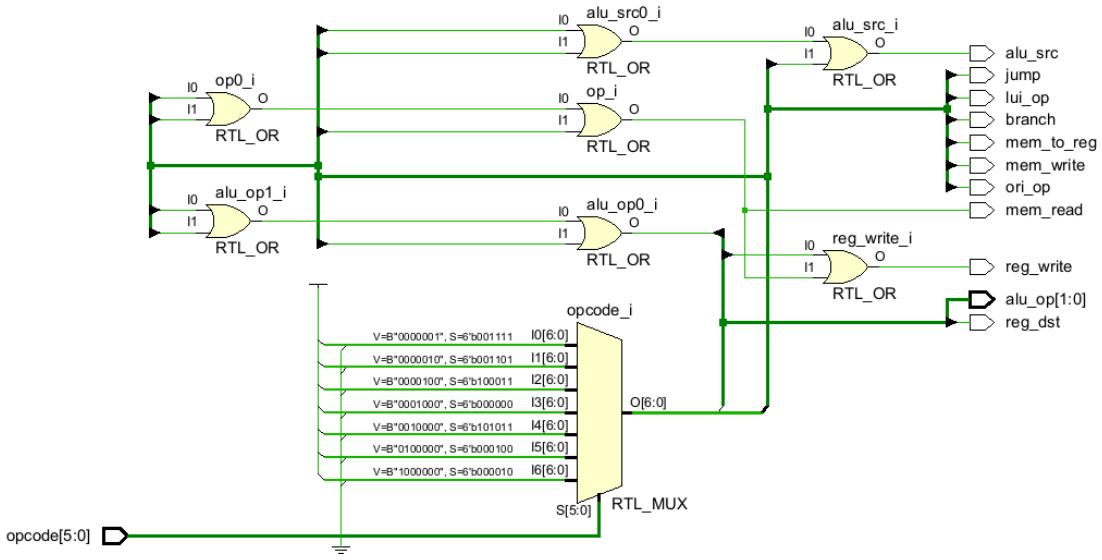
111550136 計算機組織 lab2 report

1. ALU control



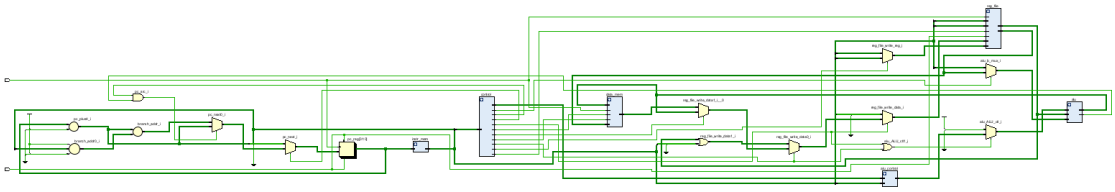
找出 operation 的 4 個 bits 分別是怎麼由 funct 跟 alu_op 組成的，分別 assign 4 個 bit 即可。

main control

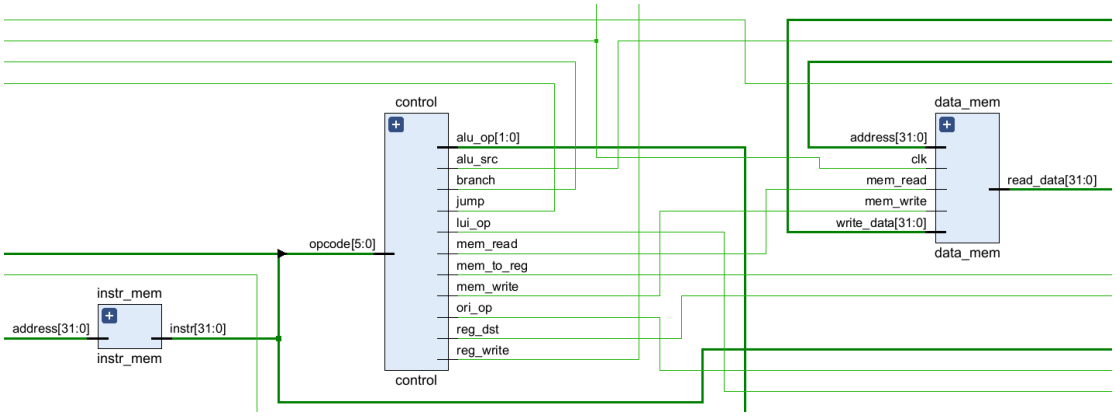


一樣把每一個 output 是由哪些 op_code 時才會有值的寫出來，分別 assign 就好，有另外加入 jump, lui_op, ori_op

single-cycle processor

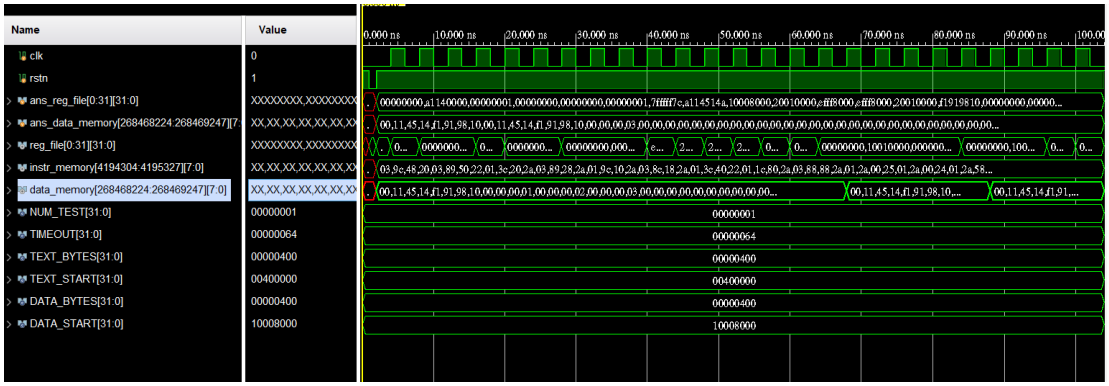


圖太大無法全部放大，我把中間的部分放大上上來



圖有點複雜，依照 6 個 step 分別接好線跟加入 jump,lw,sw,ori,lui 的線即可

2. waveform



Pass 測資

```
==== Test  0 RUNNING ====
number of instructions: 29 , exit @ 0x00400074
run 25 cycles
==== Test  0 PASSED ====
#### Test Result ####
Passed  1 : 0
Failed  0 :
#### all passed!
```

我沒有自己寫其他測資來測試我的 code

3.

(1). write 在 posedge clk 時會寫入新的 pc 值，去更新下一個指令，read register 則是在 decode 的時候會根據指令給值，read memory 則是在存取記憶體階段，會設定 data_mem_mem_read，讓 data_mem_read_data 可以被使用。

(2).blt:

blt $rs, $rt, offset

slt $at, $rs, $rt

bne $at, $zero, offset

bgt:

bgt $rs, $rt, offset

slt $at, $rt, $rs

bne $at, $zero, offset

ble:

ble $rs, $rt, offset

slt $at, $rt, $rs

beq $at, $zero, offset

bge:

bge $rs, $rt, offset

slt $at, $rs, $rt

beq $at, $zero, offset

(3).例如　loop: beq $zero, $zero, loop;　這樣就會一直在 loop 裡面無限跑

(4).

```
#  假設 target address 0xXXXXXXXX

lui $t0, 0xXXXX    #將 target address [31:16]存入$t0

ori $t0, $t0, 0xXXXX    #將 target address[15:0]存入$t0

srl $t1, $t0, 28    #右移 28bit,將[31:28]存入$t1

# increment the upper 4 bits to move to the next block

addi $t1, $t1, 1    #Increment the upper 4 bits in $t1

sll $t1, $t1, 28    #Shift left logical by 28 bits to move the upper 4 bits back

ori $t0, $t1, $t0 # OR the incremented upper 4 bits with the original lower 28 bits

j $t0 # Jump to the reconstructed target address in the next block
```

這樣可以用 j 去完成跳到 next block 的要求

(5). 因在每一個 clock 中只能執行一條指令，這樣效率非常差，另外他要求要在一個 clock cycle 中完成所有的指令，這樣週期無法過短，進而限制頻率的提升，導致速度較慢，另外整體的複雜度也比 Pipelined designs 高出不少。

4. 這次 lab 遇到比較大問題的是在 single cycle 中，常常跑出錯誤的結果，因為少接一兩條線，或者接錯線，我改了很多次，也請 g 老師看了很多次，最後才 pass 測資。