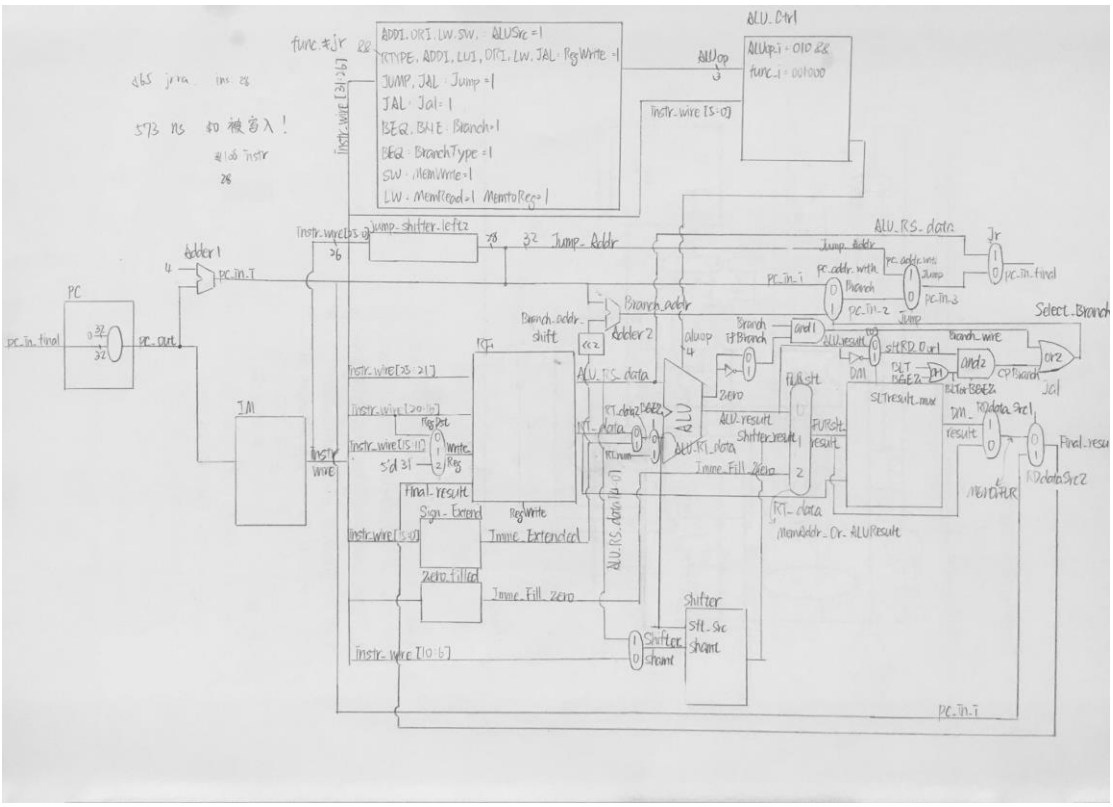
Computer Organization

Architecture diagrams:



Hardware module analysis:

Data_Memory(clk_i, addr_i, data_i, MemRead_i, MemWrite_i, data_o);

MemRead_i :決定 MEM 是否讀取(lw 使用)

MemWrite_i:決定 MEM 是否寫入(sw 使用)

MEM[addr_i]:讀取位置 addr_i 的 MEM 內容

Data_o:輸出 MEM[addr_i]

MUX ALU_src2Src2:

0: RT data

1:0

Select 1 用來執行 BGEZ

MUX ALU_zero
0:zero
1:~zero
輸出: ifBranch

And and1:
輸出 Branch_wire = Branch & ifBranch

MUX SLTresult_mux
0:ALU 執行完 slt 後的 bit 0 之結果
1: ALU 執行完 slt 後的 bit 0 結果的 inverse
輸出:sltRD_0or1

or or1
輸出 BLT | BGEZ
and and2
輸出 CMBranch = sltRD_0or1 & (BLT | BGEZ)

or or2
輸出 select_Branch = Branch_wire | CPBranch

MUX Pc_addr_with_Branch:
0:PC+4
1:Branch address
控制信號線: select_Branch

MUX pc_addr_with_jump:
0: Pc_addr_with_Branch 輸出之 address
1: Jump address
控制信號線: jump

MUX pc_addr_with_jr:
0: pc_addr_with_jump 輸出之 address
1: $ra
控制信號線:jr

Finished part:

Basic instruction
Advanced set 1
Advanced set 2

Problems you met and solutions:

Jr $ra 指令不用寫入 register RD，找了很久才找到問題

Summary:

無