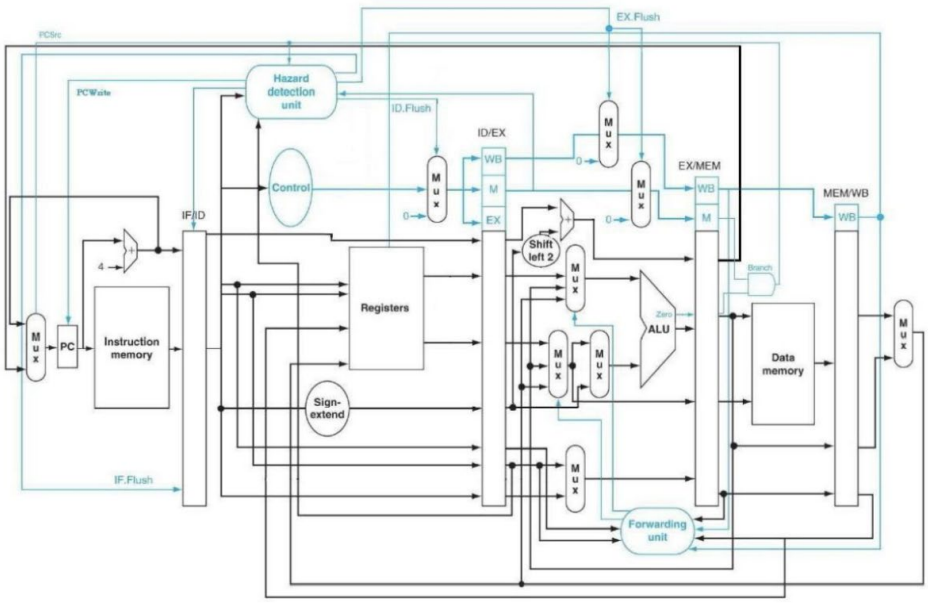
Architecture



Hardware Module Analysis

兩個新的 module

Forwarding(

RS_addr_EX_i,

RT_addr_EX_i,

RD_addr_MEM_i,

RD_addr_WB_i,

regWrite_MEM_i,

regWrite_WB_i,

Forward_A_o,

Forward_B_o

);

負責 Forwarding 的訊號：

- EX hazard
  - if (EX/MEM.RegWrite and (EX/MEM.RegisterRd ≠ 0)
    and (EX/MEM.RegisterRd = ID/EX.RegisterRs))
    ForwardA = 10
  - if (EX/MEM.RegWrite and (EX/MEM.RegisterRd ≠ 0)
    and (EX/MEM.RegisterRd = ID/EX.RegisterRt))
    ForwardB = 10
- MEM hazard
  - if (MEM/WB.RegWrite and (MEM/WB.RegisterRd ≠ 0)
    and (MEM/WB.RegisterRd = ID/EX.RegisterRs))
    ForwardA = 01
  - if (MEM/WB.RegWrite and (MEM/WB.RegisterRd ≠ 0)
    and (MEM/WB.RegisterRd = ID/EX.RegisterRt))
    ForwardB = 01

```
Hazard_Detection(
    PC_src_i,
    MemRead_EX_i,
    RS_IFID_addr_i,
    RT_IFID_addr_i,
    RT_IDEX_addr_i,
    PC_write_o,
    IF_wait_o,
    IF_flush_o,
    ID_flush_o,
    EX_flush_o
    );
```

如果 branch 要跳，就要把 PC_write_o 設成 1，IF_wait_o 設成 0， IF_flush_o、ID_flush_o、EX_flush_o 設成 1。

如果是 load 而且出現 data hazard，就要把 PC_write_o 設成 0，IF_wait_o 設成 1，ID_flush_o 設成 1， IF_flush_o、EX_flush_o 設成 0。

其他正常情況則是 PC_write_o 設成 1，其他都設成 0。

ProgramCounter -> 增加 input : pc_write_i

pc_write_i 是 1 的時候才讀新的 program counter

Pipe_Reg -> 增加 input: flush_i、wait_i

flush_i 是 1 的時候 output 都給 0

wait_i 是 1 的時候不讀新的

Decoder -> 增加 output : BranchType_o

控制 beq、bne、bge、bgt，訊號會一路傳到 MEM 的 stage 去

Pipe_Reg #(.size(155)) ID_EX

增加 input: ins_mem_ID_w[25:21]、BranchType_w

增加 output: RS_addr_EX_w、BranchType_EX_w

Pipe_Reg #(.size(109)) EX_MEM

增加 input: BranchType_EX_w

增加 output: BranchType_MEM_w

再用一個 MUX_4to1 決定是否要跳

MUX_4to1 #(.size(1)) Mux_branch(

```
    .data0_i(ALU_zero_MEM_w),
    .data1_i(~ALU_zero_MEM_w),
    .data2_i(~ALU_result_MEM_w[31]),
    .data3_i(~(ALU_zero_MEM_w | ALU_result_MEM_w[31])),
    .select_i(BranchType_MEM_w),
    .data_o(Branch_result_w)
);
```

最後跟 branch 的訊號(是否為 1)做 and，算出 PC_source 的 input：

```
bz_w = M_Ctrl_MEM_w[2] & Branch_result_w;
```

Problems and Solutions

一開始沒把 pc_write 加進 ProgramCounter 裡面所以 r6 是 0，其他都對，後來有發現

Result

```
Register==============================================================

r0=         0, r1=        16, r2=       256, r3=         8, r4=        16, r5=         8, r6=        24, r7=        26

r8=         8, r9=         1, r10=         0, r11=         0, r12=         0, r13=         0, r14=         0, r15=         0

r16=         0, r17=         0, r18=         0, r19=         0, r20=         0, r21=         0, r22=         0, r23=         0

r24=         0, r25=         0, r26=         0, r27=         0, r28=         0, r29=         0, r30=         0, r31=         0


Memory================================================================

m0=         0, m1=        16, m2=         0, m3=         0, m4=         0, m5=         0, m6=         0, m7=         0

m8=         0, m9=         0, m10=         0, m11=         0, m12=         0, m13=         0, m14=         0, m15=         0

r16=         0, m17=         0, m18=         0, m19=         0, m20=         0, m21=         0, m22=         0, m23=         0

m24=         0, m25=         0, m26=         0, m27=         0, m28=         0, m29=         0, m30=         0, m31=         0
** VVP Stop(0) **
** Flushing output streams.
** Current simulation time is 210000 ticks.
>
```

Summary

在 hazard 的判斷和不同種類的 branch 判斷上想了比較久，然後這次要改的東西也比較多，所以檢查了很久看有沒有錯或是少改。