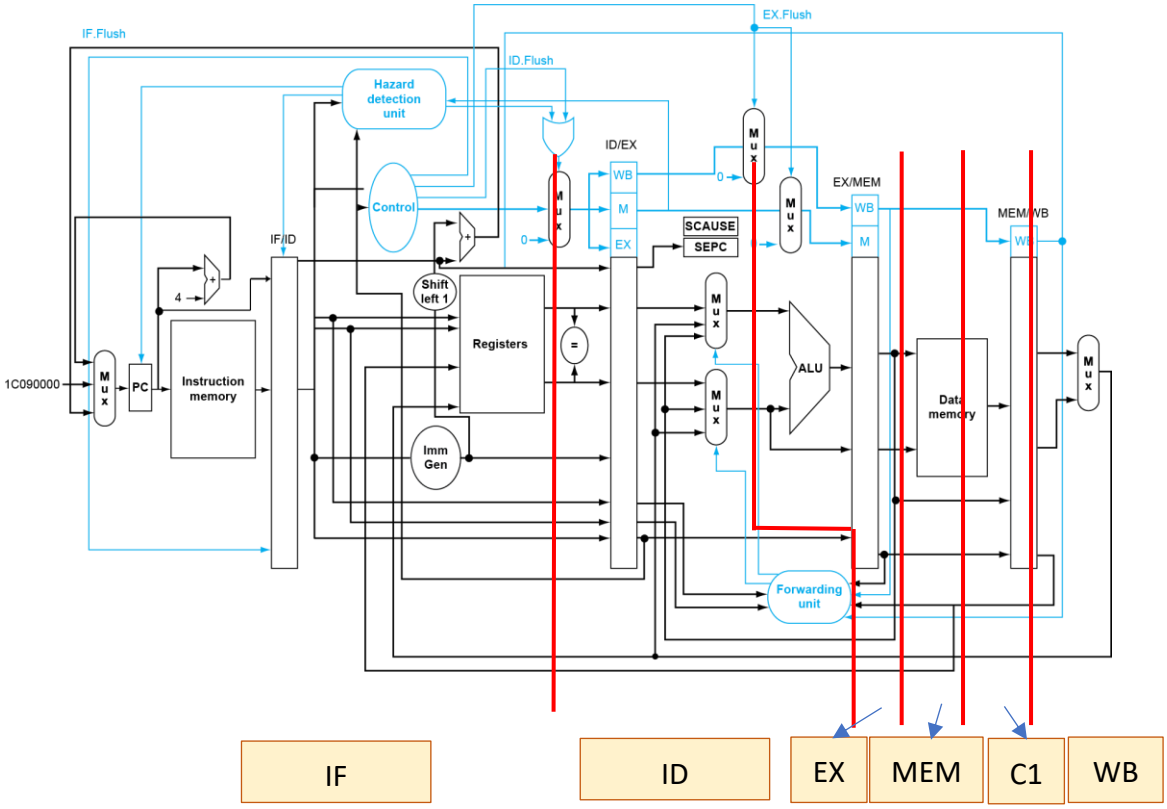
Pipeline Report

- 架構(紅線為 cpu.v 中切割 stage 的方式):



1. IF: 取出 rs1、rs2、imm 的值，設定 control 與 ALU control。
2. ID: 選擇出 EX stage 中 ALU 的兩個 input(包含取得最新的 rs1、rs2)。
3. EX: 以 ID stage 取得的兩個 ALU input 進入 ALU 運算(依據 ALU control 選擇要做甚麼運算)，將計算出的值傳給 o_w/r_addr_w。計算 branch address(pc + imm(signed)<<1)。
4. MEM: 依據 EX 算出的值，可知 branch 是否成立，在這個 stage 中對 o_i_addr_w、c_pc_w(next current pc)、pc_w(fetch instruction pc)賦值。
5. C1: 為了等 load 的值出來而設計的 stage。

   load behavior:

o_r_addr_w =	addr1	addr2	addr3	addr4
i_d_valid_data=	0	0	0	1
data =	x	not yet	not yet	ok

   中間要等 2 個 cycle，第三個 cycle 才會取到讀好的 data。
6. WB: 把 ALU result 或 load 的 data 存入 register 中。

- Pipeline registers(在 cpu.v 中的命名以及其用途):

  IFID: [0]instruction, [1]PC, [2]PC+4, [3]rs1, [4]rs2, [5]imm, [6]control, [7]ALU control

  IDEX: [0]ALU up input, [1]ALU down input, [2]read data 2, [3]control, [4]PC, [5]PC+4, [6]instruction, [7]ALU control, [8]imm

  EXMEM: [0]ALU zero, [1]ALU result, [2] read data 2, [3] control, [4]PC, [5]PC+4, [6]branch address, [7]instruction

  MEMC1: [0]read data, [1]ALU result, [2]control, [3]instruction

  C1WB: [0]read data, [1]ALU result, [2]control, [3]instruction

- What is the latency of each module in your design? (e.g. ALU, register file)

  Critical path

```
ABC: Path 56 --   30135 : 3    1 AOI21_X1 A =   1.06  Df =2378.6  -26.2 ps  S =  13.6 ps
ABC: Start-point = pi5731 (\IDEX_r_reg[8][1]).  End-point = po3335 (\EXMEM_w[6] [63]).
ABC: + write_blif <abc-temp-dir>/output.blif
```

  是計算 branch address 的部分: EXMEM_w[6] (branch address)= pc + imm<<1

- Which path is the critical path of your cpu? And how can you decrease the latency of it?

  Critical path 同上題，減少的辦法是可以先將 imm*2 在上個 stage 做完。

- Data/Control Hazard & forwarding:

  1. 為了避免 Data Hazard，一旦 load 指令進入 IF stage 中，往後的 3 個 cycle 都輸入 nop。雖然 load 在第三個 cycle 就把值讀好了，但為了不讀取 instruction.v 接續吐出的 3 個 instruction，所以 nop 的數目為 3。
  2. 為了避免 Control Hazard，當 branch 指令進入時，往後 6 個 cycle 都設為 nop，前 3 個 nop 是為了等 EX stage ALU 減法的結果，確定是否該 branch，以及將下一個要 fetch 的指令位置設為 branch target address。後 3 個 nop 則是等 instruction 在 3 個 cycle 後被 fetch 出來而設計。
  3. 經過 1. 2. 的處理，在 ID stage 中只需考慮 forwarding，不會有值還沒 load 好或是 branch 還沒確定而計算出錯誤的值這些 hazard 發生。
  4. Forwarding 在 ID stage 做，因為 EX stage 要用”最新的”值做 ALU 運算，因此我的作法是在前一個 stage 就決定好 ALU 的 2 個 input 數值。同上課時的說法，從 pipeline 暫存空間中取出還沒存進 register 但已經算好的數值，其優先序位是 IDEX>EXMEM>MEMC1>C1WB。

- Describe 3 different workloads attributes, and which one can be improved tremendously by branch predictor?
  Workload1 有很多 add/sub/xor/or/and 指令
  Workload2 有很多 bne/beq
  Workload3 有很多 add/sub/xor/or/and 指令
  Workload2 can be improved tremendously by branch predictor

- Is it always beneficial to insert multiple stage of pipeline in designs? How does it affect the latency?
  分太多也不好，forwarding 控制會變更麻煩。適當的切割 stage，一個 cycle 需要做的是減少則 latency 下降。

- 執行步驟紀錄:
  $make
  $cp ../cpu_syn.v cpu_syn.v　　把 cpu_syn.v 複製到 codes 資料夾下
  $make test　　將 Makefile 裡的 test 改為 cpu_modified.f 檢查是否正確與 cycle 數
  $yosys -l cpu.yslog -q cpu_modified.ys 檢查有沒有 dlatch，從 cpu.yslog 看出 time 與 area 資訊

- 結果紀錄:

Cycle number and correctness:

```
root@bced96459536:~# make test
iverilog -D T1 -f cpu_modified.f
./testbench.v:149: warning: parameter ADDR_W not found in test_cpu.u_cpu.
./testbench.v:148: warning: parameter DATA_W not found in test_cpu.u_cpu.
./testbench.v:147: warning: parameter INST_W not found in test_cpu.u_cpu.
vvp ./a.out
VCD info: dumpfile cpu.vcd opened for output.
Store test
Check memory
Correct!
Cycle count:          19
iverilog -D T2 -f cpu_modified.f
./testbench.v:149: warning: parameter ADDR_W not found in test_cpu.u_cpu.
./testbench.v:148: warning: parameter DATA_W not found in test_cpu.u_cpu.
./testbench.v:147: warning: parameter INST_W not found in test_cpu.u_cpu.
vvp ./a.out
VCD info: dumpfile cpu.vcd opened for output.
Load test
Check memory
Correct!
Cycle count:          27
iverilog -D T3 -f cpu_modified.f
./testbench.v:149: warning: parameter ADDR_W not found in test_cpu.u_cpu.
./testbench.v:148: warning: parameter DATA_W not found in test_cpu.u_cpu.
./testbench.v:147: warning: parameter INST_W not found in test_cpu.u_cpu.
vvp ./a.out
VCD info: dumpfile cpu.vcd opened for output.
Add sub test
Check memory
Correct!
Cycle count:          19
```

```
iverilog -D T4 -f cpu_modified.f
./testbench.v:149: warning: parameter ADDR_W not found in test_cpu.u_cpu.
./testbench.v:148: warning: parameter DATA_W not found in test_cpu.u_cpu.
./testbench.v:147: warning: parameter INST_W not found in test_cpu.u_cpu.
vvp ./a.out
VCD info: dumpfile cpu.vcd opened for output.
And or xor test
Check memory
Correct!
Cycle count:          27
iverilog -D T5 -f cpu_modified.f
./testbench.v:149: warning: parameter ADDR_W not found in test_cpu.u_cpu.
./testbench.v:148: warning: parameter DATA_W not found in test_cpu.u_cpu.
./testbench.v:147: warning: parameter INST_W not found in test_cpu.u_cpu.
vvp ./a.out
VCD info: dumpfile cpu.vcd opened for output.
Andi ori xori test
Check memory
Correct!
Cycle count:          21
iverilog -D T6 -f cpu_modified.f
./testbench.v:149: warning: parameter ADDR_W not found in test_cpu.u_cpu.
./testbench.v:148: warning: parameter DATA_W not found in test_cpu.u_cpu.
./testbench.v:147: warning: parameter INST_W not found in test_cpu.u_cpu.
vvp ./a.out
VCD info: dumpfile cpu.vcd opened for output.
Slli srli test
Check memory
Correct!
Cycle count:          19
```

```
iverilog -D T7 -f cpu_modified.f
./testbench.v:149: warning: parameter ADDR_W not found in test_cpu.u_cpu.
./testbench.v:148: warning: parameter DATA_W not found in test_cpu.u_cpu.
./testbench.v:147: warning: parameter INST_W not found in test_cpu.u_cpu.
vvp ./a.out
VCD info: dumpfile cpu.vcd opened for output.
Bne beq test
Check memory
Correct!
Cycle count:          36
iverilog -D T8 -f cpu_modified.f
./testbench.v:149: warning: parameter ADDR_W not found in test_cpu.u_cpu.
./testbench.v:148: warning: parameter DATA_W not found in test_cpu.u_cpu.
./testbench.v:147: warning: parameter INST_W not found in test_cpu.u_cpu.
vvp ./a.out
VCD info: dumpfile cpu.vcd opened for output.
Workload1 test
Check memory
Correct!
Cycle count:        5421
iverilog -D T9 -f cpu_modified.f
./testbench.v:149: warning: parameter ADDR_W not found in test_cpu.u_cpu.
./testbench.v:148: warning: parameter DATA_W not found in test_cpu.u_cpu.
./testbench.v:147: warning: parameter INST_W not found in test_cpu.u_cpu.
vvp ./a.out
VCD info: dumpfile cpu.vcd opened for output.
Workload2 test
Check memory
Correct!
Cycle count:       19538
iverilog -D T10 -f cpu_modified.f
./testbench.v:149: warning: parameter ADDR_W not found in test_cpu.u_cpu.
./testbench.v:148: warning: parameter DATA_W not found in test_cpu.u_cpu.
./testbench.v:147: warning: parameter INST_W not found in test_cpu.u_cpu.
vvp ./a.out
VCD info: dumpfile cpu.vcd opened for output.
Workload3 test
Check memory
Correct!
Cycle count:       15257
```

Make cpu.yslog and no error output

```
root@bced96459536:~/codes# cp ../cpu_syn.v cpu_syn.v
root@bced96459536:~/codes# cd ..
root@bced96459536:~# yosys -l cpu.yslog -q cpu_modified.ys
root@bced96459536:~# ls
```

Check time and area

```
root@bced96459536:~# make time
cat cpu.yslog | grep WireLoad
ABC: WireLoad = "none"  Gates =  20725 ( 15.9 %)   Cap =  3.5 ff (  1.7 %)
Area =    23602.18 ( 89.6 %)   Delay =  2234.57 ps  (  1.6 %)
```