Homework 4

Problem 1

Suppose the CPU is pipelined with Data Forwarding, there are five instructions:

```
irmovl $0, %eax
popl %eax
popl %ebx
addl %eax,%ebx
pushl %eax
```

Draw the pipelined diagram of these instructions, and point out forwarding(s) in the diagram.

1) Initially, all registers have value $0.

2) has to explicitly specify the *src* and *dest* of each forwarding value (e_valE, W_valE, m_valM, etc.)

(MEM[%esp]=1,MEM[%esp+4]=2)

Solution

	1	2	3	4	5	6	7	8	9
Popl %eax	F	D	E	M	W				
Popl %ebx		F	D	E	M	W			
bubble					E	M	W		
Addl %eax, %ebx			F	D	D	E	M	W	
Pushl %eax				F	F	D	E	M	W

Forwarding:
Cycle 3:
e_ValE → d_ValA; %esp
e_ValE → d_ValB; %esp
Cycle 5:
m_ValM → d_ValB; (2)
Cycle 6:
e_ValE → d_ValA;(3)
w_ValE → d_ValB; %esp

Problem 2

```
#demo-retB.ys
0x000:      irmovl Stack,%esp   # Intialize stack pointer
0x006:      call p              # Procedure call
0x00b:      irmovl $5,%esi      # Return point
0x011:      halt
0x020: .pos 0x20
0x020: p: irmovl $-1,%edi       # procedure
0x026:      ret
0x027:      irmovl $1,%eax      # Should not be executed
0x02d:      irmovl $2,%ecx      # Should not be executed
0x033:      irmovl $3,%edx      # Should not be executed
0x039:      irmovl $4,%ebx      # Should not be executed
0x100: .pos 0x100
0x100: Stack:                   # Stack: Stack pointer
```

In the above example, when **ret** at 0x026 is in its D, E, M stages, there would be **stall** at F stage and **bubble** at D stage. Explain why this happens and the difference between stall and bubble?

Solution

A pipeline stall is when a pipeline stage's input memory remains the same from one cycle to the next. A pipeline bubble is when a pipeline stage's input memory changes to that of a nop instruction, i.e., an instruction that does nothing. On any cycle when an instruction stalls in one stage, a bubble is injected into the subsequent stage.

Problem 3

For each of the instruction **that would cause exception**, specify its cause (halt, bad address or invalid instruction) and the stage where exception can be detected, otherwise leave blank.

Instruction	Cause	Stage
halt	Halt	F
.byte FF	Invalid instruction	F
jmp $-1	Invalid instruction	F
rmmovl %eax,0x10000(%eax) where %eax=$100	Bad address	M
xor %eax, %eax jne t t: .byte 0xFF	-	-

Problem 4

Suppose we have the following statistics for a certain program:

1) Fraction of instructions that are loads: 10%
2) Fraction of load instructions requiring stall: 20%
3) Number of bubbles injected each time: 1
4) Fraction of instructions that are cond. Jumps: 35%
5) Fraction of cond. jumps mispredicted: 40%
6) Number of bubbles injected each time: 2
7) Fraction of instructions that are returns: 1%
8) Number of bubbles injected each time: 3

Calculate CPI.

Solution

CPI=1+0.1*0.2*1+0.35*0.4*2+0.01*3=1.33