Colin Fitzsimons

Question 2

a.

Firstly the same thing happens to both instructions in the ID (instruction decode) stage of the pipeline. The instruction is taken from the instruction register and broken down into opcodes. From there it is sent onto the EX stage where the instruction is actually executed. In the case of the dsub instruction the pipeline registers would be taken up as follows:

A: R2

B: R1

imm:

And the instruction would be carried out in the ALU and passed onto the next step. In the case of the load instruction, the ALU generates the address that needs to be loaded from. The pipeline registers might look as follows:

A: r3

B: r4

Imm: 8

The ALU will calculate the address and output it to the MEM stage where it can be used to find the correct memory location to read from.

b.

In this case it would be faster to carry out three shifts and then a subtract. This would save us up to 3 cycles. A sample code segment is as follows:

```
.data
N:  .word 34
.text
ld r1, N(r0)
dsrl r2, r1, 4
dsub r1, r2, r1
HALT
```

In this case I belive when fully run we should get 9/10 cycles with a RAW stall. If we were to replace it with a similar program using a dmul instruction:

```
.data
N:  .word 34
.text
ld r1, N(r0)
daddi r2, r0, 15
dmul r1, r2, r1
HALT
```

We should now end up with cycles around 12/13 which is a good three or four cycles slower. So in this case I would say that using shifts and an addition produces a faster calculation.