A

If "li" command has op code "001" then: `RR $r` is the same with `li <($r register address = 2^2) + (funct = 2)>` = `li 6`

B

A single R-Format opcode "contains" (can be used) 4 commands due to the "funct" which is 2 bits long.

We can't have more than 4 r-format commands as we have no more available bits

So the commands will be:

> op=000 funct=00
>
> op=000 funct=01
>
> op=000 funct=10
>
> op=000 funct=11

C

0---- ----\ \ \ 1------ \AND -----+--------- I-Format / / | -----\ 2---- /-----/ +---- \ NOT \ ------ R-Format /-----/ 3

4

5

6

7

D

R-Format Commands:

> op=000 funct=00
>
> op=000 funct=01
>
> op=000 funct=10
>
> op=000 funct=11
>
> op=001 funct=00
>
> op=001 funct=01

> op=001 funct=10
>
> op=001 funct=11 The reason is because `beq` and `bne` are R-format commands already, so adding another immediate would be out of the 32 bit limit. NOTES: i = $16, j = $17, $at = $1

i

beq $16, $17, L1

ii

bne $16, $17, L1

iii

slt $at, $17, $16 bne $at, $0, L1

iv

beq $17, $16, L1 slt $at, $17, $16 bne $at, $0, L1

v

slt $at, $16, $17 bne $at, $0, L1

vi

beq $16, $17, L1 slt $at, $16, $17 bne $at, $0, L1

vii

addi $at, $0, CONST beq $16, $at, L1

viii

addi $at, $0, CONST bne $16, $at, L1

ix

slti $at, $16, CONST bne $at, $0, L1

x

addi $at, $0, CONST beq $16, $at, L1 slti $at, $16, CONST bne $at, $0, L1

xi

slti $at, $16, CONST beq $at, $0, L1

xii

addi $at, $0, CONST beq $16, $at, L1 slti $at, $16, CONST beq $at, $0, L1 addi $s3, $0, -1 Loop: addi $s3, $s3, 1 sll $t1, $s3, 2 add $t1, $t1, $s6 lw $t0, 0($t1) bne $t0, $s5, Loop Exit: ...

Commands executed with old code: 1 initilization 6*9 on each loop 4 on last loop = 59 commands

Commands executed with new code: 1 initilization 5*10 on each loop = 51 commands