Name: Janari Morrison Box: 2520 Date: 9/11/2020

HW4

1. (6 points) Convert the following C code to MIPS assembly instructions. Use the minimum number of instructions necessary. Assume that variables f, g and h are 32- bit integers stored in registers `$t0`, `$t1` and `$t2` respectively and that the base address of arrays A and B are in registers `$s3` and `$s4` respectively. A and B are arrays of 4-byte integers (this is important). If you need to store temporary values, use one of the other t registers.

a. `f = 1; A[f] = 0;`

```
ori $t0, $0, 1      # f=1
sll $t1, $t0, 2     # multiply f by four and store in t1
addi $s3, $t1, $s3  # move array pointer to the address of f
sw $0, 0($s3)       # store 0 in the array at address f
```

b. `B[1] = A[f-5];`

```
lw $t2, 4($s4)      # get B[1]
sll $t1, $t0, 2     # store f*4 in $t1
addi $t1, $t1, -20  # shift left by 5 (4*20)
addi $s3, $t1, $s3  # shift array pointer to $t1 (f-5)
sw $t2, 0($s3)      # store B[1] in A[f-5]
```

2. (4 points) Show the hexadecimal representation of the following MIPS instructions.

a. `addi $s0, $s0, 4`

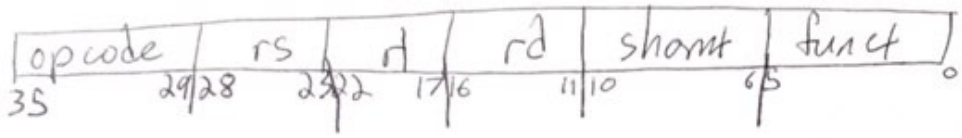
0010 0010 0001 0000 0000 0000 0000 0100

b. `lw $s1, 8($t0)`

1000 1110 0011 0001 0000 0000 0000 1000

3. (5 points) Consider changing the MIPS instruction set to support 64 registers instead of 32. Assuming no changes are made to the other instruction fields, draw the new R-type instruction format. Be sure to label each field and include its size.

The size of rs, rt, and rd would increase by 1 each ~~(all to 5 from 4)~~ (all to 6 from 5)



4. (5 points) How many instructions does MIPS support if opcode `000000` is the only opcode used for R-types? Remember that R-types also use a function code.

This means there would be 111111 ~~opcodes~~ function codes, or ~~63~~ 64 (including 0).

MIPS would support up to 64 R-type commands