Ahmad Aldasouqi
02-19-2025

Written Homework 2

2.1) The given C code is equivalent to the following mips assembly.

```
addi $s2, $s2, -5
add $s0, $s1, $s2
```

2.2) The instruction can be translated into the following C statement.

```
f = i + (g + h);
```

2.3) The C statement is equivalent to the following assembly

```
sub $t0, $s3, $s4
sll $t0, $t0, 2
add $t1, $s6, $t0
lw $t2, 0($t1)
sw $t2, 32($s7)
```

2.6) Mips systems use big-endian byte ordering, so the number already had the MSB first. The decimal form 0xab cd ef 12 is 2882400018.

2.7) The C statement is equivalent to the following assembly.

```
sll $t0, $s3, 3
add $t1, $s6, $t0
ld  $t2, 0($t1)

sll $t3, $s4, 3
add $t4, $s6, $t3
ld  $t5, 0($t4)

add $t6, $t2, $t5
sd  $t6, 64($s7)
```

2.8)

```
A[1] = (int) &A[0];
f = (int) &A[0] + (int) &A[0];
```

Ahmad Aldasouqi
02-19-2025

2.12) To decode the instruction, we first must match the bits with their respective fields.
[0x0, 0x10, 0x10, 0x10, 0x00,0x20]
000000 10000 10000 10000 00000 100000
The op field is 0, so it is R-type. This can be decoded into the following instruction: add 0x10, 0x10, 0x10

2.19) The not pseudo instruction's behavior can be implemented simply by using the nor instruction with an operand of zero. This can be done with the following instruction.

```
nor $t1, $t1, $zero
```