Brian Loughran
Intro to Computer Architecture
Johns Hopkins
Module 3 Homework

2.3, 2.4

2.9

2.14, 2.15, 2.16, 2.17

2.19, all Note: Typo error in 2.19.1... the shift value is **4**

3.20, 3.21, 3.22, 3.29

2.3 [5] <§§2.2, 2.3> For the following C statement, what is the corresponding MIPS assembly code? Assume that the variables `f`, `g`, `h`, `i`, and `j` are assigned to registers `$s0`, `$s1`, `$s2`, `$s3`, and `$s4`, respectively. Assume that the base address of the arrays `A` and `B` are in registers `$s6` and `$s7`, respectively.

```
B[8] = A[i-j];
```

```
sub $t0, $s3, $s4    # I - j
lw $t1, $t0($s6)     # $t1 = A[i-j]
sw $t1, 32($s7)      # B[8] = A[i-j]
```

2.4 [5] <§§2.2, 2.3> For the MIPS assembly instructions below, what is the corresponding C statement? Assume that the variables `f`, `g`, `h`, `i`, and `j` are assigned to registers `$s0`, `$s1`, `$s2`, `$s3`, and `$s4`, respectively. Assume that the base address of the arrays A and B are in registers `$s6` and `$s7`, respectively.

```
sll $t0, $s0, 2       # $t0 = f * 4
add $t0, $s6, $t0     # $t0 = &A[f]
sll $t1, $s1, 2       # $t1 = g * 4
add $t1, $s7, $t1     # $t1 = &B[g]
lw $s0, 0($t0)        # f = A[f]
addi $t2, $t0, 4      # $t2 = &A[f + 1]
lw $t0, 0($t2)        # $t0 = A[f + 1]
add $t0, $t0, $s0     # $t0 = A[f + 1] + A[f]
sw $t0, 0($t1)        # B[g] = A[f + 1] + A[f];
```

B[g] = A[f + 1] + A[f]
f = A[f]

Brian Loughran
Intro to Computer Architecture
Johns Hopkins
Module 3 Homework

2.9 [5] <§§2.2, 2.3> Translate the following C code to MIPS. Assume that the variables `f`, `g`, `h`, `i`, and `j` are assigned to registers `$s0`, `$s1`, `$s2`, `$s3`, and `$s4`, respectively. Assume that the base address of the arrays `A` and `B` are in registers `$s6` and `$s7`, respectively. Assume that the elements of the arrays `A` and `B` are 4-byte words:

```
B[8] = A[i] + A[j];
```

```
lw $t0, $s3($s6)      # $t0 = A[i]
lw $t1, $s4($s6)      # $t1 = A[j]
add $t2, $t0, $t1     # $t2 = $t0 + $t1
sw $t2, 3($s7)        # B[8] = $t2
```

NOTE: Feedback to learner was following snip, but that looks to be a different problem than shows in the book that I have, which is restated above the solution:

2.9
```
sll   $t0, $s1, 2   # $t0 <-- 4*g
add   $t0, $t0, $s7 # $t0 <-- Addr(B[g])
lw    $t0, 0($t0)   # $t0 <-- B[g]
addi  $t0, $t0, 1   # $t0 <-- B[g]+1
sll   $t0, $t0, 2   # $t0 <-- 4*(B[g]+1) = Addr(A[B[g]+1])
lw    $s0, 0($t0)   # f   <-- A[B[g]+1]
```

2.14 [5] <§§2.2, 2.5> Provide the type and assembly language instruction for the following binary value: `0000 0010 0001 0000 1000 0000 0010 0000`

op = 000000 – 0 - add
rs = 10000 – 17 – $s1
rt = 10000 – 17 - $s1
rd = 10000 – 17 - $s1
shamt = 00000 -0
func = 100000 – 32

Type: R-Type
add $s0, $s0, $s0

2.15 [5] <§§2.2, 2.5> Provide the type and hexadecimal representation of following instruction: `sw $t1, 32($t2)`

I-Type
101011 01010 01001 00000 00000 1000000
ad49 0040 (hex)

Brian Loughran
Intro to Computer Architecture
Johns Hopkins
Module 3 Homework

2.16 [5] <§2.5> Provide the type, assembly language instruction, and binary representation of instruction described by the following MIPS fields:
op=0, rs=3, rt=2, rd=3, shamt=0, funct=34

R-Type
Binary: 000000 00011 00010 00010 00000 110010
Instruction: sub $v1, $v0, $v1

2.17 [5] <§2.5> Provide the type, assembly language instruction, and binary representation of instruction described by the following MIPS fields:
op=0x23, rs=1, rt=2, const=0x4

I-Type
Binary: 100011 00001 00010 00000 00000 00100
lw $v0, 4($at)

2.19 Assume the following register contents:
$t0 = 0xAAAAAAAA, $t1 = 0x12345678
2.19.1 [5] <§2.6> For the register values shown above, what is the value of $t2 for the following sequence of instructions?
sll $t2, $t0, 4
or $t2, $t2, $t1

$t0 = 10101010101010101010101010101010
$t1 = 00010010001101000101011001111000
run instruction 1:
$t1 = 00010010001101000101011001111000
$t2 = 10101010101010101010101010100000
run instruction 2:
$t2 = 10111010101111101111111011111000

2.19.2 [5] <§2.6> For the register values shown above, what is the value of $t2 for the following sequence of instructions?
sll $t2, $t0, 4
andi $t2, $t2, -1

$t0 = 10101010101010101010101010101010
$t1 = 00010010001101000101011001111000
run instruction 1:
$t1 = 00010010001101000101011001111000
$t2 = 10101010101010101010101010100000
run instruction 2:
$t2 = 10101010101010101010101010100000

2.19.3 [5] <§2.6> For the register values shown above, what is the value of $t2 for the following sequence of instructions?

Brian Loughran
Intro to Computer Architecture
Johns Hopkins
Module 3 Homework

```
srl $t2, $t0, 3
andi $t2, $t2, 0xFFEF
```

$t0 = 10101010101010101010101010101010
$t1 = 00010010001101000101011001111000
run instruction 1:
$t2 = 00010101010101010101010101010101
0xFFFF = 00000000000000000111111111111111
run instruction 2:
$t2 = 00000000000000000101010001010101

3.20 [5] <§3.5> What decimal number does the bit pattern 0×0C000000 represent if it is a two's complement integer? An unsigned integer?

0x0C000000 = 0000 1100 0000 0000 0000 0000 0000 0000

2's compliment: 201326592
Unsigned: 201326592

3.21 [10] <§3.5> If the bit pattern 0×0C000000 is placed into the Instruction Register, what MIPS instruction will be executed?

0x0C000000 = 0000 1100 0000 0000 0000 0000 0000 0000
jal 0

3.22 [10] <§3.5> What decimal number does the bit pattern 0×0C000000 represent if it is a floating point number? Use the IEEE 754 standard.

0x0C000000 = 0000 1100 0000 0000 0000 0000 0000 0000
Sign = 0 -> +
Exponent = 00011000 -> -127 + 24 = -103
Fraction = 00000000000000000000000 = 1 + 0 = 1
Float = 1*2*-103

3.23 [10] <§3.5> Write down the binary representation of the decimal number 63.25 assuming the IEEE 754 single precision format.

Sign = + -> 0
63.25/2^5 = 1.9765625
Exponent = 5 = -127 + 132 = 10000100
Decimal = .9765625 = 11111010000000000000
IEEE 754 = 01000010011111010000000000000000

NOTE: Feedback says: "3.23: $2.6546875\ 10^{1\,(-4)}$". Not sure what that means.

Brian Loughran
Intro to Computer Architecture
Johns Hopkins
Module 3 Homework

Double checking with an online converter tool gives the same answer I gave: