2.3

Find: B[8] = A[i-j] in MIPS where

$$f = \$s0 \; g = \$s1 \; h = \$s2 \; i = \$s3 \; j = \$s4$$
$$A = \$s6 \; B = \$s7$$

Answer:

1. sub $t0, $s3, $s4 # i-j to $t0
2. sll $t0, $t0, 2 # shift $t0 left two bits, effectively multiplying by four to get correct byte address
3. add $t0, $t0, $s6 # add (i-j) to base of A
4. lw $t1, 0($t0) # load A[i-j] to temp register
5. sw $t1, 32($s7) # store A[i-j] in B[8] address

2.4

Find: Convert MIPS instructions to C statement where

$$f = \$s0 \; g = \$s1 \; h = \$s2 \; i = \$s3 \; j = \$s4$$
$$A = \$s6 \; B = \$s7$$

Answer:

1. $t0 = f * 4
2. $t0 = &A[f]
3. $t1 = g * 4
4. $t1 = &B[g]
5. f = A[f]
6. $t2 = &A[f+1]
7. $t0 = A[f+1]
8. $t0 = A[f+1] + A[f]
9. B[g] = A[f+1] + A[f]

2.12

$s0 = 0x80000000 $s1 = 0xD0000000

(1) Find: $t0 in 'add $t0, $s0, $s1'

Answer:

$$\begin{array}{r} 1000\ 0000\ 0000\ 0000\ 0000\ 0000\ 0000\ 0000 \\ +\ 1101\ 0000\ 0000\ 0000\ 0000\ 0000\ 0000\ 0000 \\ \hline 1\ 0101\ 0000\ 0000\ 0000\ 0000\ 0000\ 0000\ 0000 \end{array}$$

$= 0x150000000$

(2) Find: Has there been overflow?

Answer: There has been overflow if register is 32 bits.

(3) Find: $t0 in 'sub $t0, $s0, $s1'

Answer: There are two methods to accomplish this subtraction - either provides the same result.

1. a - b = -(b - a)
2. a - b = a + -b

Using the first method, one can do

$$\begin{array}{r} 1101\ 0000\ 0000\ 0000\ 0000\ 0000\ 0000\ 0000 \\ -\ 1000\ 0000\ 0000\ 0000\ 0000\ 0000\ 0000\ 0000 \\ \hline 0101\ 0000\ 0000\ 0000\ 0000\ 0000\ 0000\ 0000 \end{array}$$

then simply convert using two's complement to

$$1010\ 1111\ 1111\ 1111\ 1111\ 1111\ 1111\ 1111\ +1\ =\ 1011\ 0000\ 0000\ 0000\ 0000\ 0000\ 0000\ 0000$$

$= 0xB0000000$

Using the second method, one can do

$$\begin{array}{r} 1000\ 0000\ 0000\ 0000\ 0000\ 0000\ 0000\ 0000 \\ +\ 0011\ 0000\ 0000\ 0000\ 0000\ 0000\ 0000\ 0000 \\ \hline 1011\ 0000\ 0000\ 0000\ 0000\ 0000\ 0000\ 0000 \end{array}$$

$= 0xB0000000$

(4) Find: Has there been overflow?

Answer: No, this is the desired result.

(5) Find: $t0 after 'add $t0, $s0, $s1' and 'add $t0, $t0, $s0'

Answer: From part (1) we know that $s0 + $s1 = 0x150000000. Thus, for $s0 + $t0 we can do

$$
\begin{array}{r}
\quad 1000\ 0000\ 0000\ 0000\ 0000\ 0000\ 0000\ 0000 \\
+\ 1\ 0101\ 0000\ 0000\ 0000\ 0000\ 0000\ 0000\ 0000 \\
\hline
\quad 1\ 1101\ 0000\ 0000\ 0000\ 0000\ 0000\ 0000\ 0000
\end{array}
$$

$= 0x1D0000000$

(6) Find: Has there been overflow?

Answer: There has been overflow, because the number requires 33 bits.

2.14

Find: Type and assembly language instruction for the binary value 0000 0010 0001 0000 1000 0000 0010 0000.

Answer: Look at the first six bits. They are 000000. Thus, this is an R-type instruction.

$$op = 000000 = 0$$
$$rs = 10000 = 16$$
$$rt = 10000 = 16$$
$$rd = 10000 = 16$$
$$shamt = 00000 = 0$$
$$funct = 100000 = 32$$

From Figure 2.5 in text, we know this corresponds to an 'add' instruction. Thus, the final answer is

$$\boxed{\text{add \$s1, \$s1, \$s1}}$$

2.15

Find: Type and hex representation of 'sw \$t1, 32(\$t2)'

Answer: According to Figure 2.5 in the text, sw (store word) is I-type (Immediate). Also accoring to Figure 2.5, the op code for sw is 43. Thus

$$op = 43 = 101011$$
$$rs = 10 = 01010$$
$$rt = 9 = 01001$$
$$address = 32 = 0000000000100000$$

Final binary representation is 1010 1101 0100 1001 0000 0000 0010 0000, which is equivalent to

$$\boxed{\text{0xAD490020}}$$

2.16

Find: Type, assembly language instruction, and binary representation of

$$op = 0$$
$$rs = 3$$
$$rt = 2$$
$$rd = 3$$
$$shamt = 0$$
$$funct = 34$$

Answer: According to Figure 2.5 in the text, op = 0 and funct = 34 corresponds to sub (substract) instruction, which is type-R (register). This is represented in binary by

000000 00011 00010 00011 00000 100010

Finally, 2 and 3 correspond to registers $v0 and $v1, respectively. Thus, the instruction is

$$\boxed{\text{sub \$v1, \$v1, \$v0}}$$

2.17

Find: Type, assembly language instruction, and binary representation of

$$op = 0x23$$
$$rs = 1$$
$$rt = 2$$
$$const = 0x4$$

Answer: 0x23 in binary is 00100011. The six rightmost bits are taken, which corresponds to 100011 = 35 = lw (load word). Thus, the type is type-I. 1 and 2 correspond to registers $at and $v0, respectively. Const 0x4 in binary is 0000 0000 0000 0100, or simply 4. Thus, the instruction and binary representation are

$$\boxed{\text{lw \$v0, 4(\$at)}}$$

$$\boxed{\text{100011 00001 00010 0000000000000100}}$$

2.19

$t0 = 0xAAAAAAAA $t1 = 0x12345678

(1) Find: Value of $t2 following 'sll $t2, $t0, 4' and 'or $t2, $t2, $t1'

Answer: Shift 0xAAAAAAAA left 4 times to give

$$\$t2 = 1010\ 1010\ 1010\ 1010\ 1010\ 1010\ 1010\ 0000$$

Next, perform the OR operation.

$$\begin{array}{r} 1010\ 1010\ 1010\ 1010\ 1010\ 1010\ 1010\ 0000 \\ OR\ 0001\ 0010\ 0011\ 0100\ 0101\ 0110\ 0111\ 1000 \\ \hline 1011\ 1010\ 1011\ 1110\ 1111\ 1110\ 1111\ 1000 \end{array}$$

$$\boxed{\$t2 = 0xBABEFEF88}$$

(2) Find: Value of $t2 following 'sll $t2, $t0, 4' and 'andi $t2, $t2, -1'

Answer: Shift 0xAAAAAAAA left 4 times to give

$$\$t2 = 1010\ 1010\ 1010\ 1010\ 1010\ 1010\ 1010\ 0000$$

Next perform the 'andi' operation on $t2 and -1. Note that AND immediate for -1 is a zero extended 0xFFFF.

$$\begin{array}{r} 1010\ 1010\ 1010\ 1010\ 1010\ 1010\ 1010\ 0000 \\ AND\ 0000\ 0000\ 0000\ 0000\ 1111\ 1111\ 1111\ 1111 \\ \hline 0000\ 0000\ 0000\ 0000\ 1010\ 1010\ 1010\ 0000 \end{array}$$

$$\boxed{\$t2 = 0x0000AAA0}$$

(3) Find: Value of $t2 following 'srl $t2, $t0, 3' and 'andi $t2, $t2, 0xFFEF'

Answer: Shift 0xAAAAAAAA right 3 times to give

$$\$t2 = 0001\ 0101\ 0101\ 0101\ 0101\ 0101\ 0101\ 0101$$

Next perform the 'andi' operation on $t2 and 0xFFEF.

$$\begin{array}{r} 0001\ 0101\ 0101\ 0101\ 0101\ 0101\ 0101\ 0101 \\ AND\ 0000\ 0000\ 0000\ 0000\ 1111\ 1111\ 1110\ 1111 \\ \hline 0000\ 0000\ 0000\ 0000\ 0101\ 0101\ 0100\ 0101 \end{array}$$

$$\boxed{\$t2 = 0x5545}$$

3.20

Find: Decimal representation of 0x0C000000 if (a) 2s complement and (b) unsigned.

Answer: The binary representation of 0x0C000000 is

$$0000\ 1100\ 0000\ 0000\ 0000\ 0000\ 0000\ 0000$$

Because the leftmost bit is zero, this is a positive number in the 2s complement system. Thus, the value is the same regardless of 2s complement or unsigned, and is equal to

$$\boxed{201326592}$$

3.21

Find: MIPS instruction represented by 0x0C000000

Answer: The binary representation of 0x0C000000 is

0000 1100 0000 0000 0000 0000 0000 0000

The first six bits representing the op are 000011. This corresponds to the 'jal' jump and link instruction.

3.22

Find: Convert 0x0C000000 to floating point value using IEEE 754 standard.

Answer: The binary representation of 0x0C000000 is

0000 1100 0000 0000 0000 0000 0000 0000

From this, we see that the sign bit S is 0, the exponent bits E are 00011000, and the mantissa bits M are 000 0000 0000 0000 0000 0000.

$$S = 0$$
$$E = 00011000 = 24$$
$$M = 000\ 0000\ 0000\ 0000\ 0000\ 0000$$

Using a bias of 127, we see that the exponent is (24-127), or -103. The form of the floating point number is

$$\begin{aligned} &(-1)^S \cdot (1 + frac) \cdot 2^E \\ &= (-1)^0 \cdot (1 + 0) \cdot 2^{-103} \\ &= 1 \cdot 2^{-103} \end{aligned}$$

$$\boxed{9.8607613\text{e-}32}$$

3.29

Find: Sum of 2.612e1 and 4.150390625e-1 assuming 16-bit half precision and 1 guard, 1 round, and 1 sticky bit.

Answer: First, convert numbers to binary. The first value is 26.125. 26 is simply 11010 and the decimal is determined by

$$0.125 \cdot 2 = 0.25$$
$$0.25 \cdot 2 = 0.5$$
$$0.5 \cdot 2 = 1.0$$

Thus, $26.125 = 11010.0010$

The second value is 0.4150390625. The decimal is determined by

$$0.4150390625 \cdot 2 = 0.830078125$$
$$0.830078125 \cdot 2 = 1.66015625$$
$$0.66015625 \cdot 2 = 1.3203125$$
$$0.3203125 \cdot 2 = 0.640625$$
$$0.640625 \cdot 2 = 1.28125$$
$$0.28125 \cdot 2 = 0.5625$$
$$0.5625 \cdot 2 = 1.125$$
$$0.125 \cdot 2 = 0.25$$
$$0.25 \cdot 2 = 0.5$$
$$0.5 \cdot 2 = 1.0$$

Thus, $0.4150390625 = 0.01101010010$.

Next, we need to adjust the smaller value to match the exponent of the larger value.

$$11010.001 = 1.1010001 \cdot 2^4$$
$$0.01101010010 = 0.00000110101001 \cdot 2^4$$

Performing the addition yields

$$\begin{array}{r} 1.10100010000000 \\ +\ 0.00000110101001 \\ \hline 1.10101000101001 \end{array}$$

In IEEE 754 half-precision, the mantissa is 10 bits, plus 3 GRS bits. Thus, the mantissa is 1010100010. The guard bit is 1, the round bit is 0, and the sticky bit is 1 because there is a 1 in a position past the twelth bit (13^{th} sticky bit or beyond), so GRS = 101 and we round up, making the mantissa 1010100011.

$$1.1010100011 \cdot 2^4 = 11010.100011 = \boxed{26.546875}$$

The IEEE 764 binary representation is calculated by the following

$$S = 0$$
$$E = 4 + 15 = 19 = 10011$$
$$M = 1010100011$$
$$\boxed{0100111010100011}$$