Assignment_ 1 Solution (with questions)

2.1) Convert the following C code to MIPS code. Assume that the variables f, g, h, and i are given and could be considered 32-bit integers.

F=g + (h-5)

A:

addi f, h, -5 (note, no subi)

add f, f, g

2.2) For the following MIPS statements, what is the corresponding C statement?

add f, g, h

add f,I, f

A:

f = g + h +i

2.5) Show how the value 0xabcdef12 would be arranged in memory of a little-endian and a big-endian machine. Assume the data is stored starting at address 0.

https://en.wikipedia.org/wiki/Endianness

"In big-endian format, whenever addressing memory or sending/storing words bytewise, the most significant byte—the byte containing the most significant bit—is stored first (has the lowest address) or sent first, then the following bytes are stored or sent in decreasing significance order, with the least significant byte—the one containing the least significant bit—stored last (having the highest address) or sent last."

A:

Big Endian

Byte 0: ab

Byte 1: cd

Byte 2: ef

Byte 3: 12

Little Endian

Byte 0: 12

Byte 1: ef

Byte 2: cd

Byte 3: ab

2.7

Translate the following C code to MIPS 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. Assume that the elements of the arrays A and B are 4-byte words:

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

```
Sll $t0,$s3, 2 # we multiply i by 4

Add $t0, $t0, $s6 #now we have the address of A[i]

Lw $t0, 0($t0) #now we load a word of A[i] into register $t0

Sll $t1, $s4,2 #we multiply j by 4

Add $t1, $t1, $s6

Lw $t1, 0($t1)

Add $t1, $t0, $t1

Sw $t1, 32($s7) # store the result to B[8]
```