Problem Set 3

Justin Ely

605.204.81.FA15 Computer Organization

22 September, 2015

2.3)

```
sub $t0, $s3, $s4     #$t0 holds (i-j)
lw $t1, $t0($s6)      #$t1 holds A[i-j]
sw $t1, 8($s7)        #$sets A[i-j] at B[8]
```

2.7)

Hex	A	B	C	D	E	F	1	2
Bin	1010	1011	1100	1101	1110	1111	0001	0010

The little endian and big endian representations are reverses of each other as shown below.

address	0	1	2	3	4	5	6	7
Big Endian	0100	1000	1111	0111	1011	0011	1101	0101
Little Endian	1010	1011	1100	1101	1110	1111	0001	0010

2.9)

```
sll $t0, $s3, 2
sll $t1, $s4, 2
lw $t2, $t1($s6)
lw $t3, $t1($s6)

add $t4, $t2, $t3
sw $t4, 32($s7)
```

2.14)

op code	source register	second source	dest. register	shift amount	function
000000	10000	10000	10000	00000	100000
0	16	16	16	0	32

0 for the op code makes this an R-type instruction which corresponds to the assembly code:

```
add $s0, $s0, $s0
```

2.15)

```
sw $t1, 32($t2)
```

The $t1 register's number code is 9_{10}, and the $t2 register's number is 10_{10}. The sw (store word) op code is 43_{10}, which makes this and I-type instruction.

op code	source register	destination	constant or address
101011	01001	01010	0000000000100000
43	9	10	32

Converting this binary representation to hexadecimal gives the value: AD2A0020

2.16)

op code	source register	second source	dest. register	shift amount	function
000000	00011	00010	00011	00000	100010
0	3	2	3	0	34
R-type	$v1	$v0	$v1	0	sub

The op code of 0 identifies this as an R-type instruction, and the function value of 34_{10} identifies this as subtraction. The assembly instruction is then:

```
sub $v1, $v0, $v1
```

3.17)

Op code $23_{hex} = 35_{10} =$ lw. Register 1 is $at, and register 2 is $v0. Op code of lw identifies this as an I-type instruction, so the cons is the address for the last 16 bits of the instruction. The binary representation and assembly instruction is then:

op code	source register	destination	constant or address
1000011	00010	00001	0000000000000100
43	9	10	32
lw	$at	$v0	4

```
lw $v0, 4($at)
```

2.19)