Mingkuan Pang

1.

```
lw  $t0, 16($s1)
lw  $t1, 36($s1)
sw  $t0, 36($s1)
sw  $t1, 16($s1)
```

2.

$s1: A1 B2 C3 D4

1010 0001 1011 0010 1100 0011 1101 0100

sar 3

1111 0100 0011 0110 0101 1000 0111 1010

F 4 3 6 5 8 7 A

new $t1: F436587A

1111 0100 0011 0110 0101 1000 0111 1010

slr 1

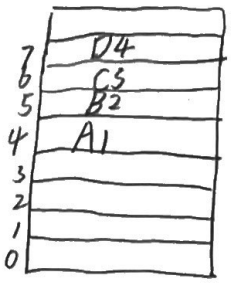
0111 1010 0001 1011 0010 1100 0011 1101

7 A 1 B 2 C 3 D

new $t2: 7A1B2C3D

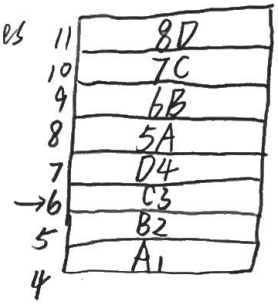
3.

sw $s1, 4($zero) 1 mem access



sw $s2, 8($zero) 1 mem access

lw $s1, 6($zero) 2 mem accesses



$s1: C3 D4 5A 6B

There are 1+1+2=4 memory accesses are made.

4. (a)

add $t1, $s2, $s1	type: R	addressing mode: Register addressing
lw $t0, 4($t1)	type: I	addressing mode: Base/displacement
bne $t0, $s5, End	type: I	addressing mode: PC-relative
addi $s1, $s1, 2	type: I	addressing mode: Immediate
subi $s1, $s1, 1	type: I	addressing mode: Immediate
j Start	type: J	addressing mode: direct addressing

4. (b)

35	9	8	4
100011	01000	01001	00000000 0000 0100
8	D	0	9	0	0	0	4

lw $t0, 4($t1) : 8D090004

(c)

5	8	21	1
00101	01000	10101	0000 0000 0000 0001
1	5	1	5	0	0	0	1

bne $t0, $t5, End : 15150001

5. The addressing mode used in Jump instruction is direct addressing mode.

The direct addressing is a scheme in which the address specifies which memory word or register contains the operand.

In Jump instruction PC will execute the instruction which is on the address that stored in operand if the condition is satisfied.

like j start. PC will execute the instruction which is stored in "start". Because "start" is actually a variable name for an address. That is direct addressing