

2.3

```
sub $t0, $s3, $s4
sll $t0, $t0, 2
add $t0, $S6, $t0
lw $t1, 0($t0)
sw $t1, 32($s7)
```

2.4

$B[g] = A[f] + A[l+f]$

2.7

Little-Endian		Big-Endian	
Address	Data	Address	Data
12	ab	12	12
8	cd	8	ef
4	ef	4	cd
0	12	0	ab

2.12

2.12.1 50000000

2.12.2 overflow

2.12.3 B0000000

2.12.4 no overflow

2.12.5 D0000000

2.12.6 overflow

2.19

2.19.1 0xBABEFEF8 Note: 1st line code should be "sll \$t2, \$t0, 4"

2.19.2 0xAAAAAAAA0

2.19.3 0x00005545

2.20 srl \$t0, \$t0, 11
sll \$t0, \$t0, 26
ori \$t2, \$0, 0x03ff
sll \$t2, \$t2, 16
ori \$t2, \$t2, 0xffff
and \$t1, \$t1, \$t2
or \$t1, \$t1, \$t0

```

2.27    addi $t0, $0, 0
          beq  $0, $0, TEST1
LOOP1:   addi $t1, $0, 0
          beq  $0, $0, TEST2
LOOP2:   add  $t3, $t0, $t1
          sll  $t2, $t1, 4
          add  $t2, $t2, $s2
          sw   $t3, ($t2)
          addi $t1, $t1, 1
TEST2:   slt   $t2, $t1, $s1
          bne  $t2, $0, LOOP2
          addi $t0, $t0, 1
TEST1:   slt   $t2, $t0, $s0
          bne  $t2, $0, LOOP1

```

```

2.31 fib:   addi $sp, $sp, -12      # make room on stack
            sw    $ra, 8($sp)      # push $ra
            sw    $s0, 4($sp)      # push $s0
            sw    $a0, 0($sp)      # push $a0 (N)
            bgt   $a0, $0, test2   # if n>0, test if n=1
            add   $v0, $0, $0      # else fib(0) = 0
            j     rtn              #
test2:      addi $t0, $0, 1         #
            bne   $t0, $a0, gen     # if n>1, gen
            add   $v0, $0, $t0      # else fib(1) = 1
            j     rtn
gen:        subi $a0, $a0, 1        # n-1
            jal   fib              # call fib(n-1)
            add   $s0, $v0, $0      # copy fib(n-1)
            sub   $a0, $a0, 1      # n-2
            jal   fib              # call fib(n-2)
            add   $v0, $v0, $s0     # fib(n-1)+fib(n-2)
rtn:        lw    $a0, 0($sp)       # pop $a0
            lw    $s0, 4($sp)       # pop $s0
            lw    $ra, 8($sp)       # pop $ra
            addi  $sp, $sp, 12      # restore sp
            jr    $ra

```

```

# fib(0) = 12 instructions, fib(1) = 14 instructions,
# fib(N) = 26 + 18N instructions for N >= 2

```

```
2.34 f: addi $sp,$sp,-12
        sw    $ra,8($sp)
        sw    $s1,4($sp)
        sw    $s0,0($sp)
        move  $s1,$a2
        move  $s0,$a3
        jal   func
        move  $a0,$v0
        add   $a1,$s0,$s1
        jal   func
        lw    $ra,8($sp)
        lw    $s1,4($sp)
        lw    $s0,0($sp)
        addi  $sp,$sp,12
        jr    $ra
```

2.39 Generally, all solutions are similar:

```
lui $t1, top_16_bits  
ori $t1, $t1, bottom_16_bits
```

2.47

2.47.1 2.6

2.47.2 0.88

2.47.3 0.5333333333