Yu xinmiao 于欣淼 518021910792

# VE370 Homework 1

## Q1

$$\textbf{CPU Time} = \frac{\text{IC} \times \text{CPI}}{\text{Clock Rate}} = \frac{100000 \times 1.5}{500\text{MHz}} = \frac{IC_B \times 2.3}{850\text{MHz}}$$

Then from the equation, calculate that $IC_B \approx 110870$.

## Q2

$$\textbf{CPU Time} = \frac{\text{IC} \times \text{CPI}}{\text{Clock Rate}} = \frac{1100 \times 1 + (200 + 150) \times 7 + 120 \times 3}{2 \times 10^9} = 1.955 \times 10^{-6}\ [s]$$

$$\textbf{CPI} = \frac{\text{CPU time} \times \text{Clock Rate}}{IC} = \frac{1955 \times 2}{1570} \approx 2.49$$

## Q3

1)

$$\textbf{CPU Time} = \frac{\text{IC} \times \text{CPI}}{\text{Clock Rate}} = \frac{2560 \times 1 + (1280 \times 4) + (256 \times 2)}{3 \times 10^9} = 2.73 \times 10^-6\ [s]$$

2) $\textbf{CPU Time} = \dfrac{1280 \times 1 + (640 \times 6) + (128 \times 2)}{3 \times 10^9} \approx 1.79 \times 10^{-6}\ [s]$

3) $\textbf{CPU Time} = \dfrac{640 \times 1 + (320 \times 8) + (64 \times 2)}{3 \times 10^9} \approx 1.11 \times 10^{-6}\ [s]$

4) $\textbf{CPU Time} = \dfrac{320 \times 1 + (160 \times 10) + (32 \times 2)}{24 \times 10^9} \approx 6.61 \times 10^{-7}\ [s]$

## Q4

x = x - y + z -72

```
add   $t0, $s0, $s2    # $t0 = x + z
addi  $t1, $s2, 72     # $t1 = y + 72
sub   $s0, $t0, $t1    # $s0 = $t0 - $t1
```

## Q5

B[8] = i + A[j]

```
sll   $t2, $t1, 2      # $t2 = $t1 * 4 = 4j
add   $t2, $t2, $s5    # $t2 = address of A[j]
lw    $t3, 0($t2)      # load from memory to register, $t3 = A[j]
add   $t3, $t3, $t0    # $t3 = i + A[j]
sw    $t3, 32($s6)     # store the result from the register to memory
```

## Q6

Assume `a, b, c, d` in `$s0, $s1, $s2, $s3` respectively. The address of array A is 0x00000100, such that A[0] = 0x000011f0

```
/*** Line by line ***/  // e:0x
temp0 = &A[1]           // temp0: 0x00000104  *temp0: 0x0000F1a4
temp1 = &A[0]           // temp1: 0x00000100  *temp1: 0x000011f0
A[2] = temp1            // *0x00000108 = 0x00000100
temp0 = A[1]            // temp0: 0x0000F1a4
a = temp0 + temp1       // 0x0000F1a4 + 0x00000100
/***Simplify***/
a = A[1] + &A[0]
```

The value of $s0 is 0x0000F2a4

## Q7

```
lui   $s0, 0x1000   # $s0 = 0x10000000
lb    $s2, 2($s0)
```

In `$s2` : 0x00000066

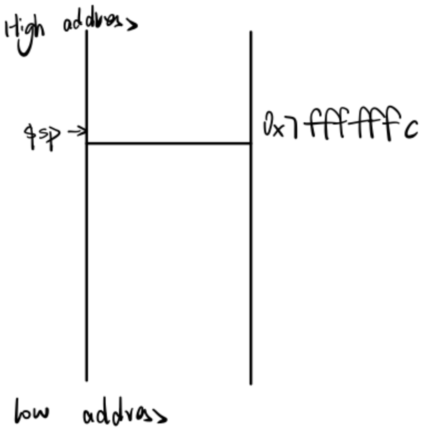
## Q8

```
slt $t2, $t0, $t1       # signed $t0 < $t1, so $t2 = 1
beq $t2, $0, ELSE       # $t2 != 0, not branch to ELSE
j DONE                  # jump to done
ELSE: addi $t2, $0, -2  # not executed
DONE: ......
```

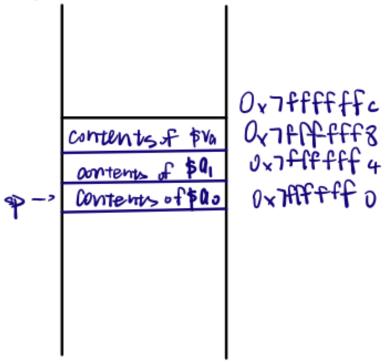
The value of `$t2` is 1.

## Q9

```
positive:
    addi $sp, $sp, -12  # adjust stack to make room for 3 items
    sw $ra, 8($sp)      # store the return address
    sw $a1, 4($sp)      # store the function arguments
    sw $a0, 0($sp)
    jal addit
    lw $a0, 0($sp)      # restore function arguments
    lw $a1, 4($sp)
    lw $ra, 8($sp)      # restore return address
    addi $sp, $sp, 12   # pop 3 items from stack
    slt $v0, $zero, $v0 # if (0 < $v0), $v0 = 1; else $v0 = 0
    jr $ra
addit:
    add $t0, $a0, $a1   # parameter variable a, b correspond to the argument register
    add $v0, $t0, $zero # copy into a return register
    jr $ra
```

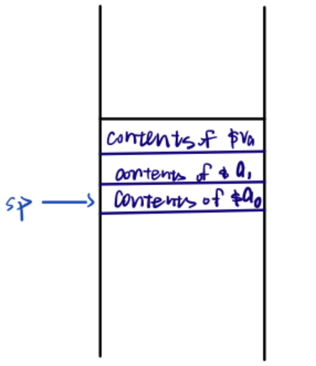


positive



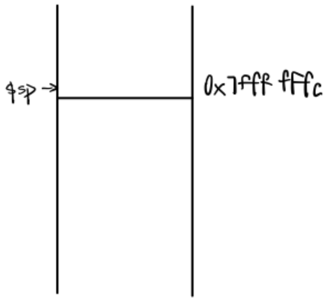
Before call addit

addit



no change in stack
when call addit

positive



In positive,
before jr $ra