VE370 HW1 Sample Solution

1. 110870 Instructions

2. 1955ns, CPI=2.49

3. 2731ns, 1792ns, 1109ns, 661ns

4. Sample answer:

```
sub $s0, $s0, $s1
add $s0, $s0, $s2
addi $s0, $s0, -72
```

5. Sample answer:

```
sll $t2, $t1, 2
add $t2, $t2, $s5
lw $t3, 0($t2)
add $t0, $t0, $t3
sw $t0, 32($s6)
```

Do not write "`lw $t0, $s5($t2)`"

6. Sample answer:

```
B[2] = B;

a = B[1] + B;

$s0=0x0000F2A4
```

7. Sample answer:

```
or $t0, $0, $0

lui $t0, 0x1000

lbu $s2, 2($t0)
```

8. `$t2 = 1`

9. Sample Answer

```
positive:  addi $sp, $sp, -4
           sw $ra, 0($sp)
           jal addit
           slt $v0, $zero, $v0
           lw $ra, 0($sp)
           addi $sp, $sp, 4
addit:     add $v0, $a0, $a1
           jr $ra
```

After calling addit:

Address	Content
0x7ffffffc	old stack
0x7ffffff8	$ra
0x7ffffff4	$a0
0x7ffffff0	$a1

VE370 HW2 Sample Solution

Problem 1

```
lui $t0, 0x0f00           # decimal equivalent is 3840
lb $s1, 2($t0)
```

Content of $s1 = 0000 0000 0000 0000 0000 0000 0100 0111 = 0x00000047

Note: there are other operations like using `addi` and `sll` to obtain the address. As long as there are no mistakes like immediate number out of range, it should be ok.

Problem 2

```
FACT: addi $sp, $sp, -8    # 8 -> -8
      sw $ra, 4($sp)
      sw $a0, 0($sp)
      add $s0, $0, $a0     # not an error, but may not be necessary here
      slti $t0, $a0, 2
      beq $t0, $0, L1
      addi $v0, $0, 1      # switch with mul
      addi $sp, $sp, 8     # -8 -> 8
      jr $ra
L1:   addi $a0, $a0, -1
      jal FACT
      mul $v0, $s0, $v0    # switch with addi
      lw $a0, 0($sp)       # switch the offsets here
      lw $ra, 4($sp)
      addi $sp, $sp, 8     # -8 -> 8
      jr $ra
```

Problem 3

1) `sub $t1, $t3, $t2`

2) R-type

Problem 4

1) 1000 1110 0101 0001 1111 1111 1110 0000

2) I-type

Problem 5

1)

op	rs	rt	rd	shamt	funct
6 bits	7 bits	7 bits	7 bits	5 bits	6 bits

Total bits: 38

Note: shamt here still does not change to 7 bits

2)

op	rs	rt	constant or address
6 bits	7 bits	7 bits	16 bits

Total bits: 36

Problem 6

1) 4 times smaller, the range is now $-2^{13} \sim 2^{13} - 1$.

2) No impact, since `jr` is **R-type**, and the address used by `jr` is fetched from the corresponding register directly.

Problem 7

Address	Instruction
0x1000f400	0000 0000 0000 1000 0110 1000 0010 1010
0x1000f404	0001 0101 1010 0000 0000 0000 0000 0001
0x1000f408	0000 1000 0000 0000 0011 1101 0000 0110
0x1000f40c	0010 0010 0111 0011 0000 0000 0000 0010
0x1000f410	0010 0101 0100 1010 0000 0000 0000 0001
0x1000f414	0000 1000 0000 0000 0011 1101 0000 0000
0x1000f418	.....

Note:

- Please pay attention the address calculation, register number, and correct order of the instruction fields.
- This time we do not give deduction to those who switch the rs, rt field incorrectly for `bne`, since one of the MIPS reference cards on Canvas (mips-ref) has a typo on branch instructions, but please note the

correct order should be `bne rs, rt, L1`.

Problem 8 & 9

Skip

Problem 10

	Text Size	0x440
	Data Size	0x90
Text	Address	Instruction
	0x00400000	lui $at, 0x1000
	0x00400004	ori $a0, $at, 0
	…	…
	0x00400140	sw $a0, 8040($gp)
	0x00400144	jmp 0x04002C0
	…	…
	0x004002C0	jal 0x0400000
	…	…
Data	0x10000000	(X)
	…	…
	0x10000040	(Y)

Note:

- Please pay attention to the dependency
- The address of data segment starts from 0x10000000
- Please pay attention to hexadecimal number calculation