IT5002 Computer Systems and Applications
Tutorial 2

1. **MIPS Bitwise Operations**

   Implement the following in MIPS assembly. Assume that integer variables **a**, **b** and **c** are mapped to registers $s0, $s1 and $s2 respectively. Each part is independent of all the other parts. **For bitwise instructions (e.g. ori, andi, etc),** any immediate values you use should be written in binary for this question. This is optional for non-bitwise instructions (e.g. addi, etc).

   Note that bit 31 is the most significant bit (MSB) on the left, and bit 0 is the least significant bit (LSB) on the right, i.e.:

MSB					LSB
Bit 31	Bit 30	Bit 29	…	Bit 1	Bit 0

   a. Set bits 2, 8, 9, 14 and 16 of **b** to 1. Leave all other bits unchanged.

   b. Copy over bits 1, 3 and 7 of **b** into **a**, without changing any other bits of **a**.

   c. Make bits 2, 4 and 8 of **c** the inverse of bits 1, 3 and 7 of **b** (i.e. if bit 1 of **b** is 0, then bit 2 of **c** should be 1; if bit 1 of **b** is 1, then bit 2 of **c** should be 0), without changing any other bits of **c**.

2. [AY2013/14 Semester 2 Exam]

The mysterious MIPS code below assumes that **$s0 is a 31-bit binary sequence**, i.e. the MSB (most significant bit) of **$s0** is assumed to be zero at the start of the code.

```
      add  $t0, $s0, $zero   # make a copy of $s0 in $t0
      lui  $t1, 0x8000
lp:   beq  $t0, $zero, e
      andi $t2, $t0, 1
      beq  $t2, $zero, s
      xor  $s0, $s0, $t1
s:    srl  $t0, $t0, 1
      j    lp
e:
```

a) For each of the following initial values in register **$s0** at the beginning of the code, give the hexadecimal value of the content in register $**s0** at the end of the code.

i. Decimal value **31**.

ii. Hexadecimal value **0x0AAAAAAA**.

b. Explain the purpose of the code in one sentence.

3. Given two integer arrays *A* and *B* with unknown number of elements, and their base addresses stored in registers **$s0** and **$s1** respectively, study the MIPS code below. Note that an integer takes up 32 bits of memory.

```
      addi $t0, $s0, 0
      addi $t1, $s1, 0
loop: lw   $t3, 0($t0)
      lw   $t4, 0($t1)
      slt  $t5, $t4, $t3      # line A
      beq  $t5, $zero, skip   # line B
      sw   $t4, 0($t0)
      sw   $t3, 0($t1)
skip: addi $t0, $t0, 4
      addi $t1, $t1, 4
      bne  $t3, $zero, loop
```

a. What is the purpose of register **$t1** in this code?

b. If array *A* = {7, 4, 1, 6, 0, 5, 9, 0} and
array *B* = {3, 4, 5, 2, 1, 0, 0, 9},

give the final content of these two arrays.

c. How many **store word** operations are performed given the contents of the arrays in part (b)?

d. What is the value (in decimal) of the immediate field in the machine code representation of the **bne** instruction?

e. The two lines indicated as "line A" and "line B" represent the translation of a MIPS pseudo-instruction. Give the corresponding pseudo-instruction.

4. You accidentally spilled coffee on your best friend's MIPS assembly code printout. Fortunately, there are enough hints for you to reconstruct the code. Fill in the missing lines (shaded cells) below to save your friendship.

Answer:

Instruction Encoding	MIPS Code
	# `$s1` stores the result, `$t0` stores a non-negative number
	`addi $s1, $zero, 0` #Inst. address is 0x00400028
`0x00084042`	`loop: srl $t0, $t0, 1`
`0x11000002`	
`0x22310001`	
	`j loop`
	`exit:`

b. Give a simple mathematic expression for the relationship between **$s1** and **$t0** as calculated in the code.