CMPE324 Midterm Sample Questions

Q1)
a) Consider the following IEEE single precision floating pointing number:

10111111011000000000000000000000

What is the corresponding decimal equivalent?

……

b) Represent the following number in single precision floating point number using IEEE754 standard: $-14.125_{10} = -1100.001_2$

……

c) Provide what number the following reserved IEEE754 codes represent:

Sign	exponent	significant	the number
0	0	0	……
1	255	0	……
0	255	non-zero	……

Q2) Using single precision floating point arithmetic, complete the following MIPS code segment that computes the volume of a sphere of radius R. Assume that the value of the radius R is placed in $f12 register. The result must be $f0 register.

```
SphereVolume:
     # load $f4 register with the value of π (3.1456)
     ……
     # compute Area= 4.0.π.R.R.R/3   let 4.0/3 be 1.3333
     ……
     ……
     ……
     ……

     j $ra
```

Q3) Order the following (single precision) floating point numbers in increasing value from 1 to 4, where 1 is the smallest number (*i.e.*, the most negative) and 4 is the largest number (*i.e.*, the most positive). IEEE754 standard is used.

_____ 0 01010011 11010011101010010101001

_____ 0 01010011 00101100010101101010110

_____ 1 11010011 00101100010101101010110

_____ 0 11010011 11010011101010010101001

Q4) Complete the following floating function that treats its argument $a0 as a 32-bit float and compute twice its absolute value in $v0.

```
TwiceAbs:
      mtc1 $a0, $f0    # $f0=$a0
      abs.s $f0, $f0   # $f0=abs($f0)
      ……..……# complete the required task
      ……..……# return result in $v0 integer register
      jr $ra
```

Q5) Consider the following MIPS code segment:

```
      addi $v0, $0, 0
Loop: andi $t0, $a0, 1
      add $v0, $v0, $t0
      srl $a0, $a0, 1
      bne $a0, $0, Loop
```

A/ Trace this code segment.

$a0	**$v0**	**? $a0 ≠ $0**
1010		

B/ Write the C++ equivalent.

……

……

……

C/ Explain in English the main job of this sequence of instructions.

Q6) Complete the following procedure that returns the maximum value of the two integer arguments $a0 and $a1. The procedure returns the maximum value in $v0 register.

```
# on entry: $a0 = a (1st argument), $a1 = b (2nd argument)
# result must go $v0
Max: move $v0, $a0   # Initially maximum is assumed to be a0
     ……..   # compare $a0 with $a1 and then
     …….. # do the necessary action
exit: jr $ra                # return with answer in $v0
```

Q7) The following MIPS function receives one non-negative argument in $a0 and returns one output in $v0, what will this function return if it is called with an argument ($a0) of 4? Show the work in the given table.

```
func:
     li $t0, 1
     li $t1, 1
loop:
     blt $a0, 2, exit
     add $t2, $t0, $t1
     move $t0, $t1
     move $t1, $t2
     addi $a0, $a0, -1
     j loop
exit:
     move $v0, $t1
     jr $ra
```

$a0	$t0	$t1	$t2	$v0

The function returns …….

Describe in English what the function actually computes? ……

Q8) Consider the following C function which returns the smallest integer in a 1-D array.

```
// Find smallest integer in an array a0 with a1 elements
int minimum(int a0[], int a1)
{     int min, s1;
   min = a0[0];//min is initialized with the first element of a0
      for (s1=1; s1<a1; s1++)
      if (a0[s1] < min) //Found an element smaller than the current min
            min = a0[s1];
      return min;}
```

Complete the following MIPS implementation of the above minimum function.

```
      minimum:
            # Save the callee-save registers $s0 and $s1 in stack
            …….
            …….
            …….
            lw $s0, 0($a0)
            li $s1, 1
      loop: # Check s1< a1
            ……., exit
            # let $t0 point to a0[s1]
            …….
            …….
            lw $t0, 0($t0)
            bgt $t0, $s0, next
            …….
      next:
            …….
            j loop
      exit:# return $s0:
            …….
            # restore the callee-save registers $s0 and $s1 from stack
            …….
            …….
            …….
            jr $ra
```

Q9) Assume that the following MIPS program is run on a 500MHz processor, with a clock cycle time of 2ns. The number of clocks per instruction is shown in the table. Assuming that $a1 register hold the value of 10, calculate the total CPU time required for executing this function.

```
func: lw $t0,0($a0)
addi $t1,$0,1
loop:bge $t1,$a1,exit
mul $t2,$t1,4
add $t2,$t2,$a0
lw $t2,0($t2)
add $t0,$t2,$0
next:addi $t1,$t1,1
j loop
exit:add $v0,$t0,$0
jr $ra
```

Instruction type	Clock per instruction
add	4
mul	10
load	5
branch, jump	3

The total number of cycles is …….
The total CPU time is …….

Q10) Consider the following machine code segment and the corresponding MIPS and C instructions. Complete the missing instructions. Note the following MIPS instructions formats:

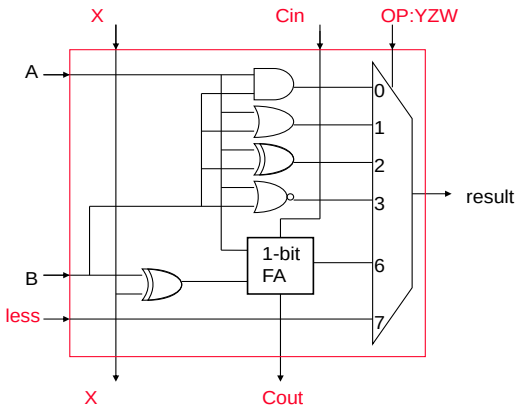
Inst. type	31-26	25-21	20-16	15-11	10-6	5-0
R	op= 0	rs	rt	rd	shamt	func
I	op= 1, 4-62	rs	rt	Immediate		
J	op= 2, 3	target address				

Address	**Machine Instruction**	**MIPS Instruction**	**C Language Instruction**
$0x00400000_{16}$	00001025_{16}	or $v0,$0,$0	**v0 = 0;**
$0x00400004_{16}$	$0005402A_{16}$	Loop: slt $t0,$0,$a1	……
$0x00400008_{16}$	11000003_{16}	……	
$0x0040000c_{16}$	00441020_{16}	add $v0,$v0,$a0	**v0 += a0;**
$0x00400010_{16}$	$20A5FFFF_{16}$	……	……
$0x00400014_{16}$	(……)$_{16}$	j Loop	
$0x00400018_{16}$		Exit:	

Show all of your work here

…… …… …… …… ……

Q11) Consider the 1-bit ALU circuit given below.
A/Fill in the given table the values of **XYZWC$_{in}$** to enable the specified ALU operation.



X	Y	Z	W	C_{in}	Operation
					AND
					OR
					XOR
					NOR
					ADD
					SUB
					Less

B/ Make the necessary connection on the circuit given below to support: slt (set less than) instruction, check if the ALU output is zero, and checking the overflow status.