CSE3038 – Spring 2021 – Midterm Exam

Q1 (10+12+7) (a,b,c are unrelated)

a) Determine the 32-bit IEEE 754 single-precision representation of −0.265625. Note that −0.265625 = −(1/4 + 1/64). Show your work, and use hexadecimal notation for your final answer.

b) Suppose in a MIPS processor $t0 contains 0xffff fffa and $t1 contains 0x0000 0007. What will be in the Hi and Lo registers after the instruction "mult $t0, $t1" is run? Show your work.

c) Give an example of two 8-bit two's complement integers such that adding them produces a carry out of the leftmost bit but not an overflow.

a) $N = -0.265625 = -(1/4 + 1/64) = -(1/2^2 + 1/2^6)$

$= -0.\underset{2^{-2}}{\underset{\downarrow}{0}1}000\underset{2^{-6}}{\underset{\downarrow}{1}}$

$N = -0.010001 = -1.0001 \times 2^{-2}$

$= (-1)^1 \times (1 + 0.0001) \times 2^{125-127}$

$125 = (01111101)_2$

1 011	1110	1 000	1 000	0 000	0 000	0000	0000
b	e	8	8	0	0	0	0

N = 0xbe880000

b) $t0 = 0xffff fffa ($t0 is negative)

$\hookrightarrow$ 1111 . . . . 1111 1010

2's Complement negative

0000 . . . 0000 0101
+ 1
―――――――――――
0000 . . . . . 00110 = 6_{10}

$-6 \times 7 = -42$

0000 . . . 0000 0010 1010 = 42
1111 . . . . 1111 1101 0101
+ 1
―――――――――――
1111 . . . . 1111 1101 0110 = −42

$\underbrace{1111}_{f}$ . . . . $\underbrace{1111}_{f}$ $\underbrace{1111}_{f}$ . . $\underbrace{1111}_{f}$ $\underbrace{1101}_{d}$ $\underbrace{0110}_{6}$

$\boxed{\begin{array}{l} Hi = 0xffff\ ffff \\ Lo = 0xffff\ ffd6 \end{array}}$

c) If the result can not be represented by hardware bits $\rightarrow$ overflow
(ex: POS+POS, NEG+NEG+ POS−NEG and NEG−POS)

No overflow $\rightarrow$ POS+NEG

```
    0000 1001     +9
 +  1111 0111     -9
 ---------------
   0000 0000
1 ←
```

```
   1111 0111    -9
   0001 0000    16
 ---------------
   0000 0111
1 ←
```

No overflow but carry out of the leftmost bit

Q2 (15+6) (a and b are unrelated)

a) Assume that i is in register $t1, n is in $t2, sum is in $t0 and base address of array arr is in $s0. Convert the following code into MIPS Code.

```
sum=0;
for (i=1; i<=n; i++) {
      if (arr[i-1] <= arr[i+1]) sum++;
}
```

b) Assuming that $a0 contains an Alphabetic character (uppercase or lowercase), write a single MIPS instruction that will make the character stored in $a0 always lower case. Note that the ASCII code of character 'A' is 0x41 while that of character 'a' is 0x61.
(Hint: one of the following instruction will help: and, or, andi, ori)

b)

```
0x61      0110  0001     Lowercase
0x41      0010  0001     Uppercase
```

Answer: ori $a0, $a0, 0x20
(If uppercase → lowercase ; if lowercase → lowercase)

a)

$$\begin{tikzcd}
i \arrow[d] & n \arrow[d] & \text{sum} \arrow[d] & \text{base addr of "arr"} \arrow[d] \\
\$t1 & \$t2 & \$t0 & \$s0
\end{tikzcd}$$

```
          add   $t0, $zero, $zero        # sum=0
          addi  $t1, $zero, 1            # i = 1
loop:     slt   $t3, $t2, $t1
          bne   $t3, $zero, exit         # if i>n exit.
          sll   $t4, $t1, 2              # 4*i
          add   $t4, $t4, $s0            # & arr [i]
          lw    $s1, -4($t4)             # $s1 = arr [i-1]
          lw    $s2, 4($t4)              # $s2 = arr [i+1]
          slt   $s3, $s2, $s1
          bne   $s3, $zero, pass-inc     # if arr [i-1] > arr[i+1]
          addi  $t0, $t0, 1
pass-inc: addi  $t1, $t1, 1
          j     loop
exit:
```

Q3 (10).
You will enhance a machine. There are two improvements: either make multiply instructions run 4 times faster than before; or make memory access instructions run 2 times faster than before.
The running time of a given program is 200msecs, where 20% is used for multiplication, 40% for memory access instructions, and 40% for other tasks.
What will be the speedup if both improvements are made?

$T^{old} = 200$ msec $\quad T^{old}_{mult} = 200 \times 0.2 = 40$ msec $\quad T^{old}_{mem} = 200 \times 0.4 = 80$ msec

$T^{old}_{others} = 80$ msec.

$T^{new}_{mult} = 40/4 = 10$ msec $\quad T^{new}_{mem} = 80/2 = 40$ msec

$T^{new}_{others} = 80$ msec.

$T^{new} = 10 + 40 + 80 = 130$ msec. $\quad \text{Speedup} = \frac{T^{old}}{T^{new}} = \frac{200}{130}$

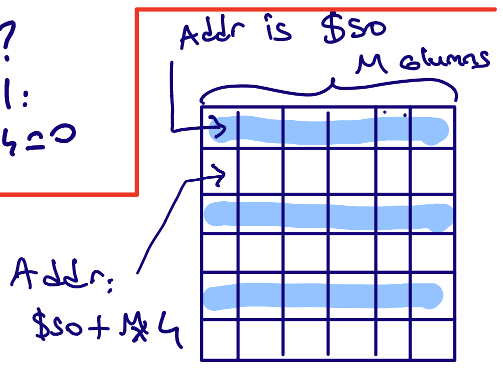
Q4 (25). Assume that array A is a square matrix of M×M integers (M rows by M columns), where the starting address of A is stored in register $s0, and M is stored in register $t0. Write a MIPS code to compute **the sum of all integers at even-numbered rows** (all integers at row 0, row 2, row 4…etc are added). (the **sum** should be in $s1).

M → $t0   Starting addr. A → $s0

#MxM matrix with indexes [i][j]

```
      sll   $t1, $t0, 2           # t1 = M * 4
      add   $s1, $zero, $zero     # sum=0
      add   $t2, $zero, $zero     # i=0
L1:   add   $t3, $zero, $zero     # J=0 (for each row)
L2:   lw    $s2, 0($s0)           #$s2 = A[i][j]
      add   $s1, $s1, $s2         # sum=sum+$s2
      addi  $t3, $t3, 1           # J++
      addi  $s0, $s0, 4           # $s0 = &A[i][j]+4
      slt   $t4, $t3, $t0         # J<M ? $t4=1 :
                                          $t4=0
      bne   $t4, $zero, L2        # if J<m continue
                                          current row

      add   $s0, $s0, $t1         #Next even row
                                     $s0=$s0+M*4
      addi  $t2, $t2, 2           # i=i+2 (even row count)
      slt   $t4, $t2, $t0         # i<M?
                                    $t4=1:
                                    $t4=0
      bne   $t4, $zero, L1
                                  #if
                                   i<M
                                   continue
```



Q5 (15). An instruction set architecture has two classes of instructions, type A and type B. There are two processors implementing the architecture, M1 (2.4Ghz) and M2 (3.2Ghz). The number of clock-cycles for the two instruction types on each processor is given as below:

	M1	**M2**
Type A	1cycle	3 cycle
Type B	2 cycle	1 cycle

We have a benchmark program which consists of both A and B type of instructions and "**F**" is equal to the fraction of instructions of *type B* used by the benchmark . The manufacturer of M2 reports their processor 4/3 times faster based on the given benchmark program. **F=?**

$$CPI_{M1} = 1 \times (1-F) + 2 \times F = 1+F$$

$$CPI_{M2} = 3 \times (1-F) + 1 \times F = 3-2F$$

$$\frac{\text{Performance}_{M2}}{\text{Perfomance}_{M1}} = \frac{\text{Exec. Time}_{M1}}{\text{Exec. Time}_{M2}} = \frac{\frac{1+F}{2.4}}{\frac{3-2F}{3.2}} = \frac{4}{3}$$

$$\frac{1+F}{3-2F} \times \frac{3.2}{2.4} = \frac{4}{3}$$

$$1+F = 3-2F$$

$$3F = 2$$

$$F = 2/3$$