Nombre: _____ Sección: _____

¡Anota tu nombre en <u>todas las hojas</u> del examen!

Tienes 2 horas para completar todos los problemas. Lee cuidadosamente todo el examen antes de empezar a trabajar. Muestra todo el trabajo conducente a tu contestación. Podrás recibir crédito parcial por contestaciones parciales siempre y cuando muestres tu trabajo por escrito. Usa tu tiempo inteligentemente. Exito!

INEL 4206 Staff

Nombre: _____ **Sección: _____**

1	25
2	25
3	25
4	25
TOTAL	100

Nombre: _____ Sección: _____

Problem 1. (25 points) Computer Arithmetic and Logic

a) (6 points) Find the shortest sequence of native MIPS instructions necessary to implement the MIPS pseudo instruction `abs $t2, $t1`. This instruction places in register `$t2` the absolute value of the two's complement integer held in register `$t1`. (HINT: it can be done in three instructions)

b) (6 points) Find the shortest sequence of native MIPS instructions to implement an `even $t2, $t1` instruction. This instruction determines if the parity of an unsigned integer number held in $t1 is even, that is if the number of bits set to 1 is even. Place a 1 or 0 in register `$t2` if the parity is even or not, respectively.

Nombre: _____ **Sección: _____**

c) (6 points) Suppose that all the conditional branch instructions except `beq` and `bne` were removed from the MIPS instructions set along with `slt` and all of its variants (`slti`, `sltu`, `sltui`). Show a sequence of MIPS instructions equivalent to the instruction `slt $t0, $s0, s1`, using the modified instruction set in which `slt` is not available. The only register that can be modified by the sequence is register `$at`.

d) (7 points) Imagine that we add a new instruction, called `div2`, that tests whether or not the content of a register is divisible by 2^n where n resides in the low 5 bits of another register. The new instruction places a 1 or 0 in a third target register if the number is or is not divisible by 2^n, respectively. Show a sequence of MIPS instructions that implements `div2 $t0, $s1, $s2` where $t0 is the target register, $s1 the number to be determined divisible, and $s2 the exponent of 2 (low 5 bits). The only register that can be modified by the sequence is register `$at`.

Nombre: _____ Sección: _____

Problem 2. (25 points) Floating Point Numbers

a) (5 points) Find the IEEE 754 single precision floating point representation of the number `3.5`:

b) (15 points) Write a sequence of MIPS instructions to multiply a single precision IEEE 754 floating point number by 2. Assume the number to be multiplied resides in register $s0. YOU CANNOT USE FLOATING POINT INSTRUCTIONS. You may use $tx registers for temporary results.

c) (5 points) What is the smallest positive single precision IEEE 754 floating point number. Remember that the binary pattern with all bits equal to zero represents the number zero. Show the binary representation of the number as well as its decimal equivalent.

Nombre: _____ **Sección: _____**

Problem 3. (25 points) Instruction Set Design

The following is MIPS code implementing a ***mystery*** function of two integer arguments ***a*** ($a0) and ***b*** ($a1) and returning an integer value in $v0. You may assume the arguments are always positive or zero.

```
mystery:    addu  $sp, $sp, -4
            sw    $ra, 0($sp)
            bgt   $a1, $zero, call
            li    $v0, zero
            j     end
call:       addi  $a1, $a1, -1
            jal   mystery
            add   $v0, $v0, $a0
end:        lw    $ra, 0($sp)
            addu  $sp, $sp, 4
            jr    $ra
```

a) (10 points) What is the output of the *mystery* when passed 5 in *a* ($a0) and 3 in *b* ($a1):

b) (5 points) Describe what the function returns as a mathematical function of its arguments *a* and *b*.

c) (10 points) Provide a C implementation of the *mystery* function

Nombre: _____ Sección: _____

Problem 4. (25 points) Easy I Simulator

The appendices include a simulator for the Easy I processor of problem set 4 written in MIPS assembly code. Consider adding a Branch on Zero Relative (`BrZr`) instruction (opcode 10 000) to the Easy I ISA. This is the same instruction that appeared in exam 3. The instruction performs the following functionality:

Indirect Bit = 0: if AC==0 then PC = PC + X + 2
Indirect Bit = 1: if AC==0 then PC = PC + MEM[X] + 2

(a) (5 points) Describe concisely how the *decode* function should be modified to incorporate the new instruction:

(b) (10 points) Show a version of the *fetchop* function that works with the new instruction:

Nombre: _____ **Sección: _____**

(c) (10 points) Provide a definition of an `fbrzr` function that executes the `BrZr` instruction:

```
fbrzr:
```

Nombre: _____ **Sección: _____**

Appendices