ENEE 350 Fall 2001: Exam 1

October 18, 9:30 - 10:45 A.M.

Name: _______________________

Note 1. If you perceive any ambiguity in any of the questions, state your assumptions explicitly.

Note 2. All work on this exam is to be wholly your own. Consulting (or copying) from other students’ answers, possessing (or copying from) unallowed material (text, notes, written papers, etc.), or aiding other students (by verbal communication or by showing your answers) will be considered a violation of the academic honor code. Violations will result in a grade of XF for the course.

1. (12 points) Give answers to each of the following questions in 4-5 sentences. If your answer contains irrelevant material, or is not to the point, you will lose points.
   a. Explain clearly the difference between unsolvable problems and intractable problems.
   b. Explain how high-level languages and assembly languages differ in the definition and handling of data types.
   c. Give an example of a function defined in the API (Application Programming Interface). Give motivations for providing that function as part of the API.
   d. Explain the differences between a pure stack-based machine, and a register-based machine.
   e. Explain the benefits of using the stored program concept.
   f. Explain how the concept of stack frames is the context of its use in implementing subroutines at the assembly language level.

2. (7 points) The MIPS assembly language defines 32 general-purpose registers, of which register R0 is hardwired to zero, and register R31 is used to store the return address when a subroutine call is made. Consider the following subset of the MIPS assembly language instructions.

   li   rt, immed ; copy immed into register rt
   la   rt, A    ; copy address of variable A into register rt
   lw   rt, A    ; copy value of variable A into register rt
   lw   rt, offset(rs); copy into register rt the value in memory location (offset + register rs)
   sw   rt, A    ; copy value in register rt to memory location corresponding to variable A
   sw   rt, offset(rs); copy the value in register rt to memory location (offset + register rs)
   add  rd, rs, rt ; add the values in registers rs and rt, and store the result in register rd
   addi rt, rs, immed ; add immed to the value in register rs, and store the result in register rt
   sub  rd, rs, rt ; subtract value in reg rt from value in reg rs, and store result in reg rd
   subi rt, rs, immed ; subtract immed from value in register rs, and store result in register rt
   mult rd, rs, rt ; multiply the values in registers rs and rt, and store result in reg rd
   and  rd, rs, rt ; and the values in registers rs and rt, and store the result in register rd
   andi rt, rs, immed ; and immed to the value in register rs, and store the result in register rt
or rd, rs, rt ; or the values in registers rs and rt, and store the result in register rd
ori rt, rs, imm ; or imm to the value in register rs, and store the result in register rt
blt rs, rt, label ; branch to address label if value in register rs is < value in register rt
bgt rs, rt, label ; branch to address label if value in register rs is > value in register rt
beq rs, rt, label ; branch to address label if values in registers rs and rt are equal
bne rs, rt, label ; branch to address label if values in registers rs and rt are not equal
j label ; jump to address label
jal label ; call subroutine at address label; store return address in register R31
jr rs ; jump to address stored in register rs; jr r31 is the return instruction
syscall ; call operating system; R2 specifies syscall type

Translate the following C program to a MIPS subset assembly language program: Comment each instruction of your code. Note that your assembly code should have the call to function smaller. To get full credit, you should mention which aregisters are used to pass parameters to smaller and the register(s) or stack offsets used to declare local variables.

```c
int a = 5;
int *b;
int c = 0;
main()
{
    int i, n;

    b = &c;
    /* Read 4 bytes from stdin, and store them in variable n */
    read(0, &n, 4);  /* This is an OS call */
    for i = 0; i < n; i++)
        if (i == i * i)
            *b = *b + smaller(a, *b);
}

int smaller(p, q)
{
    int rv;
    if (p > q)
        rv = q;
    else if (q > p)
        rv = p;
    else
        rv = 0;
    return rv;
}
```

3. (5 points) A line printer interface has an 8-bit status register (use label printer_status) and an 8-bit data register (use label printer_data). The interface works as follows. When it is ready to accept a new character for printing, the most significant bit (MSB) of its status register is set to 1,
otherwise this bit is set to 0. When a new character is to be printed, the printer device driver writes
the character to the data register of the printer interface, and sets the MSB of its status register to 0.
Write a MIPS assembly language device driver that can be used to print a string of characters using
this printer interface. The device driver is called with 2 parameters: $R4$ contains the starting address
of the string in memory, and $R5$ contains the length of the string. Notice that the characters of the
string are available in a contiguous set of memory locations. Assume program-controlled I/O. Assume
that the return address is present in $R31$.
4. (3 points) What is the decimal equivalent of the pattern $0xF0F00000$, if interpreted as per the IEEE
754 floating-point format?
5. (3 points) Design a variable-length opcode (but fixed-length instruction) to allow all of the following
to be encoded in 36-bit formats:
7 instructions with two 15-bit addresses and one 3-bit register number
500 instructions with one 15-bit addresses and one 3-bit register number
50 instructions with no addresses or registers