

# Computer Science 03-60-266 Final Examination

## Monday April 17<sup>th</sup> 2006

Dr. Alioune Ngom

Last Name:

First Name:

Student Number:

### INSTRUCTIONS

- EXAM DURATION IS 3 hours.
- OPEN NOTES EXAM: **lecture notes, textbook, . . . .**
- Calculators (including your brain and fingers) are allowed.
- READ ALL QUESTIONS CAREFULLY BEFORE ANSWERING.
- Write **all** answers and work in the space provided. Use **no** other paper.
- **WRITE YOUR FINAL ANSWERS WITH A PEN.**
- There are 6 questions (2 questions/hour). Answer **all** of them.
- There are 15 pages. Count them before you start.
- If you need to make any assumptions, state them clearly with your answers.
- Write your answer neatly. Messy work is very hard to read and may cause you to lose marks.

Q#1 (Page 2)	Q#2 (Page 4)	Q#3 (Page 6)	Q#4 (Page 7)	Q#5 (Page 10)	Q#6 (Page 12)	Total
15	10	10	30	15	20	100

**Q#1. (15)** We are given an array,  $A$ , of  $n$  integers that we wish to sum together. We can solve this summation using the following recursive algorithm

```
RecSum( $A, n$ )
  If  $n = 1$  Then
    Return  $A[0]$ ;
  Else
    Return RecSum( $A, n - 1$ ) +  $A[n - 1]$ ;
```

Write the `RecSum` function in ASM and write a piece of the main program code that calls this function. The `RecSum` function should return a value in register `EAX`. Also, assume that the *Called sub-program* cleans the stack.



**Q#2. (10 marks)** Translate the following ASM function into *pseudo-code* or *C*, *C++*, *Java*.

```
Paxen:
    PUSH EBP
    MOV  EBP, ESP
    MOV  EBX, [EBP + 12]
    MOV  EAX, [EBP + 8]
    CMP  EAX, 0
    JNE  Next
    MOV  EAX, 1
    JMP  Exit
Next:
    DEC  EAX
    PUSH EBX
    PUSH EAX
    Call Paxen ; result in EAX
    MUL  EBX
Exit:
    RET 8      ; EAX contains result
```

Piece of the program that calls Paxen is

```
:
...
PUSH x      ; a double-word
PUSH n      ; a double-word
CALL Paxen ; result in EAX
...
:
```

What does Paxen do?



**Q#3. (10 marks)** True or False? Circle the T or F, please. DO NOT just cross one out.

**T or F ?** Using ADD on two signed numbers will set or clear the OF flag, but won't change the CF flag.

**T or F ?** The SF flag will only be updated if the result of an arithmetic operation causes the number to become negative.

**T or F ?** The CF flag will be updated only for unsigned arithmetic operations.

**T or F ?** The NEG instruction will never cause an overflow condition.

**T or F ?** The SUB instruction will never overflow if both numbers are positive.

**T or F ?** For the ASM line "Var = 200", any appearance of the word Var later in the ASM file is replaced by 200.

**T or F ?** EIP can be changed in a procedure.

**T or F ?** ESP can be used to allocate space for local variables.

**T or F ?** The two-operand form of IMUL instruction will overflow if the second-half of destination operand cannot contains the product.

**T or F ?** If ECX is initialized to 0 before beginning a loop, an infinite loop will be produced.

**Q#4. (30 marks)**

1. Circle the operations that apply when the LOOPNE instruction is executed?

(May be more than one)

- a ECX is decremented
- b ECX is tested for  $\geq 0$
- c Fall through if ZF is set
- d Fall through if ZF is clear
- e ECX is tested for  $< 0$
- f Fall through if SF is clear
- g Fall through if SF is set

2. Will this jump to Target?

Circle Yes or No.

```
MOV AX, 8001h
SUB AX, 1008h
JC Target
```

3. Is *fast division* possible in ASM? Justify your answer with the example  $BX \div 36$  where  $BX = 108$ . See the example of *fast multiplication* on *Lecture 5, Page 7*.

Circle Yes or No.

4. Write the hexadecimal content of **AX** and the values of **CF** and **OF** flags, immediately after the execution of the instruction **IMUL AH**, when **AX = A98Bh**

?\_\_\_\_\_

5. Write the hexadecimal content of the quotient and remainder registers, or write **Divide Overflow**, immediately after the execution of the instruction **IDIV BL**, when **AX = A98Bh** and **BL = 8Bh**

?\_\_\_\_\_

6. Write the hexadecimal content of the destination register and the values of **CF** and **OF** flags, immediately after the execution of the instruction **ADD AL, BL**, when **AL = A9h** and **BL = 8Bh**

?\_\_\_\_\_

7. Write the hexadecimal content of the destination register and the values of **CF** and **OF** flags, immediately after the execution of the instruction **SUB AL, BL**, when **AL = A9h** and **BL = 8Bh**

?\_\_\_\_\_

8. Write the hexadecimal content of the destination register and the value of **CF** flag, immediately after the execution of the instruction **RCL AH, 3**, when **AH = ABh**

?\_\_\_\_\_



**Q#5. (15 marks)** Given the *single-precision* number -12.34

1. Give its IEEE-754 representation.  
(If the fraction does not 'stop' or does not 'repeat', then only generate '7' bits)
2. Convert the result of (1) back to decimal.  
(Use only the first '7' bits (after the 'dot') for the fraction)
3. What is the difference, if any, between the result of (2) and the original decimal number? Why such difference?

Show all your steps and clearly write your final answers



**Q#6. (20 marks)** Write an ASM program which computes the formula

$$R = -\frac{\cos 2x + 3.1(\sin y)^2}{\pi - 4z^3}$$

Your program should use all the addressing modes shown on page 11 of chapter 10. When an addressing mode is being used, please indicate the addressing mode by its initials (for instance, CSAM for "Classical Stack Addressing Mode, RPAM for "Register+Pop Addressing Mode, ...) beside the instruction that is using the addressing mode. See Lecture 10, Page 22, for instructions cos, sin and  $\pi$ . **Show all your FPU stack frames** as I have done in lab.



Space for Scrap Work

Space for Scrap Work