# HONG KONG UNIVERSITY OF SCIENCE & TECHNOLOGY

## Computer Organization (COMP 2611)

## *Spring Semester, 2012*

## Mid-term Examination 1

### March 8, 2012

### Solution

**Name:** _____ **Student ID:** _____

**Email:** _____ **Lab Section Number:** _____

**Instructions:**

1. **This examination paper consists of 12 pages in total, including 6 questions within 10 pages, 1 appendix page and 1 draft page.**
2. **Please write your name, student ID, email and lab section number on this page.**
3. **Please answer all the questions in the spaces provided on the examination paper.**
4. **Please read each question very carefully, answer clearly and to the point. Make sure that your answers are neatly written.**
5. **Keep all pages stapled together. You can tear off the appendix and draft page only.**
6. **Calculator and electronic devices are not allowed.**
7. **The examination period will last for 2 hours.**
8. **Stop writing immediately when the time is up.**

| Question | Percentage % | Scores |
|---|---|---|
| 1 | 10 | |
| 2 | 25 | |
| 3 | 20 | |
| 4 | 10 | |
| 5 | 20 | |
| 6 | 15 | |
| TOTAL | 100 | |

**Problem 1: General Knowledge (circle all correct answers) (10points)**
**Two marks per correct question.**
**If they get some correct without any wrong you give 1 point**
**And you give 0 otherwise (i.e., wrong answer or mixed-correct and wrong)**

**a) Virtually all computer designs comprise the following basic components:**

**A. Bus, memory, input/output modules, ISA**
**B. Hard disk, Monitor, Keyboard/mouse, CPU**
**C. Memory, input and output modules, Datapath, Control Unit**
**D. Memory, input modules, output modules, CPU**
**E. None of the above**

**b) An 8-input multiplexor can be controlled by an n-bit control signal. The possible value(s) of n are:**

**A. 2**
**B. 3**
**C. 9**
**D. 100**
**E. None of the above**

**c) In combinational logic:**

**A. AND and NOT gates are sufficient enough to represent any truth table**
**B. NOR gates alone are not sufficient to represent any truth table**
**C. Three two-input AND gates and two two-input OR gates are necessary to realize logic equation Y= CD + EF + G**
**D. The number of select lines in a 4x1 multiplexer is equal to the number of input lines in a 2-to-4 decoder;**
**E. None of the above**

**d) A decade counter is a sequential logic circuit that counts modulo 10 (i.e., on one decimal digit from 0 to 9) by adding one to the value currently stored in the circuit at each count and circling to 0 when it reaches 9. How many flip-flops are necessary to construct a decade counter:**

**A. 10**
**B. 3**
**C. 4**
**D. 2**
**E. None of the above**

**e) Suppose a computer represents integers in two's complement on 5 bits. Which of the following operations will yield a correct result?**

**A. 5 + 8**
**B. (-8) + (-8)**
**C. 4 – (-12)**
**D. 15 – 7**
**E. All of above**

**Problem 2: Number representation (25 points)**

a) **Convert the following numbers from base 10 to base 2, base 8, and base 16 (3points)**
**Half a point per correct answer**

$18_{10}$ = $10010_2$ = $22_8$ = $12_{16}$

$59_{10}$ = $111011_2$ = $73_8$ = $3B_{16}$

b) **In high-level languages, it is often possible to use integer variables that occupy one byte (`int8_t` in C/C++) or two bytes (`short` in C/C++) instead of 32 bits (`int` in C/C++). To convert `int8_t` or `short` into `int` we can use a cast. For example, if `i` is `int` represented on 32 bit and `j` is a short integer (16-bit) of type `short` represented on 2 bytes, we can assign `j` to `I` by `i = (int) j`, which will have the effect of expanding `j` to fit on 32 bit and yet have the same value. Give the 32-bit representation of `i` after the conversion for the following values of `j`. Please also explain why in one sentence. (3points)**
**Half a point each plus two for the explanation**
**`j` = 0000 0000 0000 1111 `i` = 0000 0000 0000 0000 0000 0000 0000 1111**

**`j` = 1000 0000 0000 1111 `i` = 1111 1111 1111 1111 1000 0000 0000 1111**

**Because of sign extension**

c) **Consider the following decimal numbers, give their 16-bit 2's complement representation (show your steps/explanation) (3points)**

i) **30 (1pt)**

**0000 0000 0001 1110**

ii) **-28 (2pt)**

**28 = 0000 0000 0001 1100**
**invert = 1111 1111 1111 1110 0011**
**add 1 = 1111 1111 1111 1110 0100**
**so -28 = 1111 1111 1111 1110 0100**

d) Consider the following 32-bit numbers in 2's complement, what decimal number do they represent (show your steps/explanation) (2points) (1 for the result and 1 for the explanation)

**1111 1111 1111 1111 1111 1111 1111 1000**

**This is a negative number; inverting and adding 1 gives its positive counterpart as 0000 0000 0000 0000 0000 0000 0000 1000 which evaluates into 8 so the sequence is -8**

e) What are the largest and smallest decimal values that can be represented in 2's complement on 8 bits? What are the largest and smallest unsigned integers that can be represented on 8 bits (2points)

Two's Complement: Largest = 127 Smallest = -128

Unsigned integer: Largest = 255 Smallest = 0

**f) Consider the following IEEE 754 single precision floating point numbers (given in hexadecimal), what values do they represent (show your steps/explanation) (5points)**

**i) 0x FF FF FF E3 (2points)**

**It is obvious here that the exponent (bits 30-23) is equal to 255 and the significand (bits 22-0) is non-zero, therefore this is NaN (not a number)**

**ii) 0x C1 60 00 00 (3points)**

**the sign bit31 = 1, exponent bits 30-23 =10000010=130 and the significand=110 0000 0000 0000 0000 0000**
**This evaluates as $-1.75 \times 2^{130-127} = -1.75 \times 8 = -14$**

**g) We want to build a floating-point number system similar to IEEE754 but on 16-bit words with 1 bit for the sign, 5 bits for the exponent, and the remaining 10 bits for the significand. Answer the following questions:**

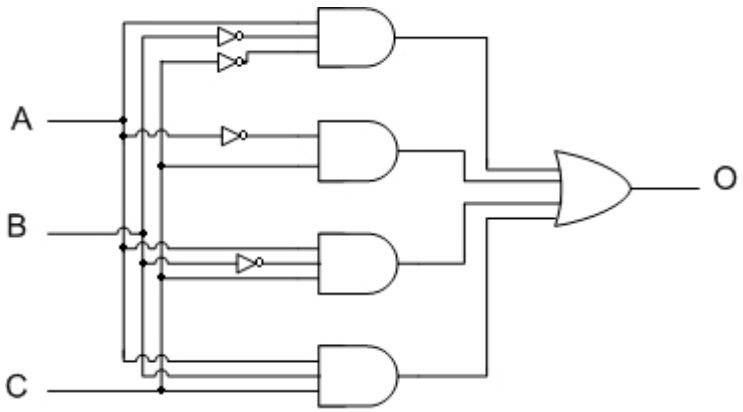
**i) What is the value of the bias (2points)**

**On five bits we can represent signed integers from -16 to 15. The bias is therefore 15**

**ii) Fill the table below to express the rules to evaluate each possible bit pattern in this floating-point number system. We will denote the sign bit by s, the biased exponent by E and the significand by F. (6points)**

| Exponent / Significand | 0 | 1-30 | 31 |
|---|---|---|---|
| 0 | 0 | $(-1)^s$ X 1.F X $2^{E-15}$ | ± ∞ |
| Non 0 | $(-1)^s$ X 0.F X $2^{-14}$ | | NAN |

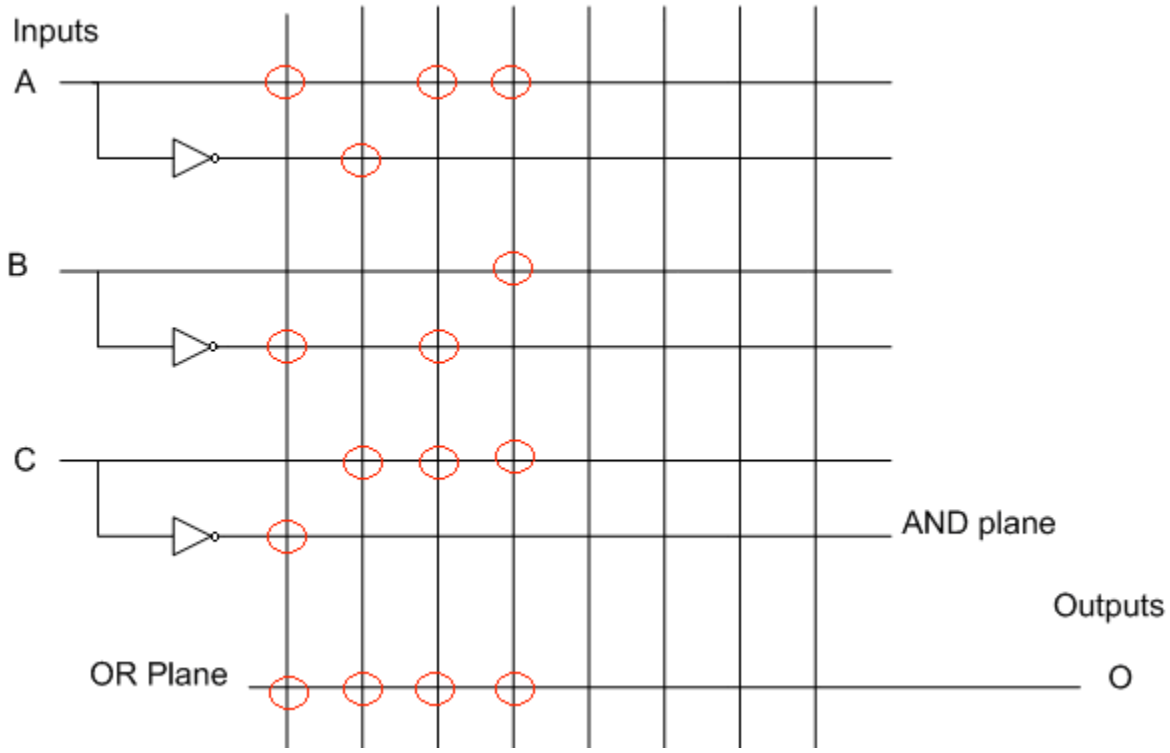
**Problem 3: Boolean algebra and Combinational Logic (20 points)**




**a) Refer to the circuit above. Give the logic equation (in Sum-of-product form) that relates the inputs (A,B,C) and the output (O) (5points)**

**O = A~B~C + ~AC + A~BC + ABC**

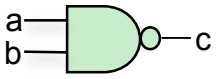
**b) Referring to (a), implement the corresponding circuit with the Programmable Logic Array circuit shown below (i.e., mark the PLA circuit appropriately in both the "OR plane" and the "AND plane"): (5points)**




**c) Using the laws of Boolean Algebra, demonstrate that the expression in (a) is equivalent to $A\overline{B}+C$. (Explain each step by listing the name of the rule you applied. The laws are provided in the appendix) (5points)**

**A~B~C + ~AC + A~BC + ABC**
**= A~B(~C+C) + C(~A+AB) distributive law**
**= A~B + C(~A+AB) complement law**
**= A~B + CA~B + C(~A+AB) absorption law**
**= A~B + C(~A+AB+A~B) distributive law**
**= A~B + C(~A+A(B+B~)) distributive law**
**= A~B + C complement law**

**d) By applying De Morgan's laws to (b), show that the circuit can be alternatively implemented using *2* NAND gates (plus some NOT gates). (hint : $\overline{\overline{A}} = A$ ) (5points)**
**(NAND is shown as follows: c is 0 only if both a and b are 1).**

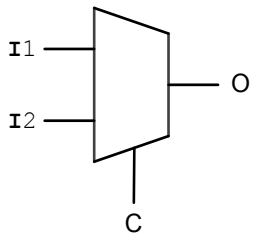



**A~B+ C**
**= ~(~(A~B+C))**
**= ~(~(A~B)~C)**
**^^^^^^**

**First NAND gate for the operation : (A NAND ~B)**
**Second NAND gate for the operation : ((A NAND ~B) NAND ~C)**

**Problem 4: Combinational Logic (Multiplexer) (10 points)**

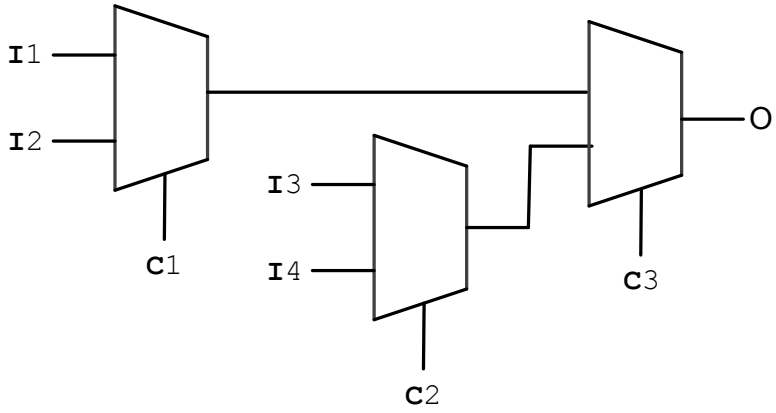
**The following figure and truth table depict an example of a 1-bit multiplexer, also known as a 2-to-1 multiplexer. If C is 0 then the output assumes the value of input `I1`, otherwise it assumes the value of input `I2`.**




| Input value to C | Output signal at O |
|---|---|
| 0 | I1 |
| 1 | I2 |

**A 4-to-1 multiplexer is a device that is capable of accepting 4 input signals (`I1`, `I2`, `I3`, and `I4`) and selecting one of them for output (O). Suppose you are asked to build a 4-to-1 multiplexer, but unfortunately you only have on hand some 2-to-1 multiplexers like the one above.**

**a) Design and draw the circuit for the 4-to-1 multiplexer using ONLY 2-to-1 multiplexors (6points)**



**b) Give the truth table for your circuit designed in a), with the inputs of the truth table being the selective inputs of all the 2-to-1 multiplexers you used and the output of the truth table is O which takes its value among `I1`, `I2`, `I3`, or `I4`. Fill as many rows and columns as necessary below. (For example, if we need two 2-to-1 multiplexers, then the inputs are: C1, C2.) (4points)**

| C1 | C2 | C3 | | Output O |
|---|---|---|---|---|
| 0 | 0 | 0 | | I1 |
| 0 | 0 | 1 | | I3 |
| 0 | 1 | 0 | | I1 |
| 0 | 1 | 1 | | I4 |
| 1 | 0 | 0 | | I2 |
| 1 | 0 | 1 | | I3 |
| 1 | 1 | 0 | | I2 |
| 1 | 1 | 1 | | I4 |
| | | | | |
| | | | | |
| | | | | |

**Problem 5: Performance (show all your steps) (20 points)**

**Consider two different implementations –M1 and M2- of the same instruction set architecture. There are four classes of instructions –A, B, C, and D- in the instruction set. The clock rates for machines M1 and M2 are 500 MHz and 750 MHz, respectively. The average numbers of cycles in each instruction class are:**

| Instruction Class | M1 CPI | M2 CPI |
|---|---|---|
| A | 1 | 2 |
| B | 2 | 2 |
| C | 3 | 4 |
| D | 4 | 4 |

**a) Define the peak performance of a machine as the fastest rate at which the machine can execute an instruction sequence chosen to maximize this rate. What is the peak performance of M1 and M2 respectively, expressed as instructions per second? (6points)**

**The fastest rate that a machine can execute a sequence is if the sequence contains all the instructions that require the least number of cpi.**
**For machine M1, it will be if all instructions are of class A, as they all require 1 cpi. With the clock rate of 500 MHz, it yields 500 million instructions per second.**
**For machine M2, it will be if all instructions are of class A, as they all require 2 cpi. With the clock rate of 750 MHz, it yields 750/2 , or 375 million instructions per second.**

**b) If the number of instructions executed in a certain program is divided equally among the classes of instructions, which machine is faster for this program? By how much? (6 points)**

**With equal distribution of instructions, the average number of cpi in M1 is given by (1+2+3+4)/4=2.5**
**The MIPS rating for M1 is: 500*10^6/2.5=200*10^6**
**The average number of cpi in M2 is given by (2+2+4+4)/4=3**
**The mips rating for M2 is: 750^6/3=250*10^6**
**M2 is faster compared to M1 by (250*10^6-200*10^6)/200*10^6=25%**

**c) If we change the instruction mix as indicated in the following table, which machine is faster, and by how much? (8points)**

| Instruction Class | M1 | M2 |
|---|---|---|
| A | 31% | 40% |
| B | 5% | 7% |
| C | 29% | 21% |
| D | 35% | 32% |

**Average CPI for M1=(0.31*1+0.05*2+0.29*3+0.35*4)/4=0.67**
**Average CPI for M2=(0.4*2+0.07*2+0.21*4+0.32*4)/4=0.76**
**MIPS rating for M1=(500*10^6/0.67)/ 10^6=746.27**
**MIPS rating for M2=(500*10^6/0.76)/10^6=657.89**

**Problem 6: MIPS Assembly Language (15 points)**

**a) Show the single MIPS instruction or shortest sequence of instructions for the following C statement, assuming that a corresponds to register $t0 and b corresponds to register $t1: (3points)**

**a = b – 17;**

**addi $t0, $t1, -17**

**b) Show the single MIPS instruction or shortest sequence of instructions for the following C statement, assuming that a corresponds to register $t0 and b corresponds to register $t1, and your solution can only use these two registers. (4points)**

**a = b * 16 + b;**

**sll $t0, $t1, 4** **Give 4 marks to everybody here because they didn't learn sll yet.**

**add $t0, $t0, $t1**

**c) Show the single MIPS instruction or shortest sequence of instructions for the following C sstatement, assuming that the integer c corresponds to register $s0 and the base address of the array of integers x is stored in register $s1. Your solution can also use any other registers. (8points)**

**x[c+1] = x[c] + c;**

**add $t0, $s0, $s0** **("sll $t0, $s0, 2" is ok for replacing these two lines)**

**add $t0, $t0, $t0**

**add $t1, $s1, $t0** **[$t1 can be replaced by any register from here down]**

**lw $t0, 0($t1)** **[ or lw $t0, ($t1) ]**

**add $t0, $t0, $s0**

**sw $t0, 4($t1)**

**APPENDIX: Boolean Algebra Laws**

**- Identity laws:**

$$A+0=A \qquad A\cdot 1=A$$

**- Annihilator (or Zero and one) laws:**

$$A+1=1 \qquad A\cdot 0=0$$

**- Complement laws:**

$$A+\overline{A}=1 \qquad A\cdot\overline{A}=0$$

**- Commutativity laws:**

$$A+B=B+A \qquad A\cdot B=B\cdot A$$

**- Associativity laws:**

$$A+(B+C)=(A+B)+C \qquad A\cdot(B\cdot C)=(A\cdot B)\cdot C$$

**- Distributivity laws:**

$$A\cdot(B+C)=(A\cdot B)+(A\cdot C) \qquad A+(B\cdot C)=(A+B)\cdot(A+C)$$

**- Idempotence:**

$$A+A=A \qquad A\cdot A=A$$

**- Absorption laws:**

$$A+(A\cdot B)=A \qquad A\cdot(A+B)=A$$

**- De Morgan Laws:**

$$\overline{A+B}=\overline{A}\cdot\overline{B}$$

$$\overline{A\cdot B}=\overline{A}+\overline{B}$$