COM219 FINAL EXAM

The exam must be completed and handed in before the end of finals. The duration of the exam is **24 hours** and starts after you acquire knowledge of any of the questions below.

Collaboration is not allowed. All work that you submit must reflect your own work. You may not give or receive any help on any of the questions in this exam. This includes the period from the time you submit your exam until the due date.

Print legibly the **time** and **date** you started the exam on the first page. Also write the **EXAM_ID** given below.

EXAM ID	22EBH44K

Sign your **honor pledge** stating that you will not collaborate, give or receive any help on this exam, and that you will not help anyone after you have submitted this exam. Any exam that does not contain the honor pledge, signature or EXAM ID will not be graded. Write the completion date and time of the exam before you submit your work. Submit it in pdf format on Moodle as soon as you finish and before the end-of-finals deadline.

START
Write the start time and exam ID on your answer sheet

Q1 (40% - 5 points each)

a) **Moore's Law**
We have seen how memory can be implemented using logic gates and know that gates are implemented using transistors. Find how much the capacity of a memory chip has increased in the last 20 years. That is, find $\textbf{capacity}_{\textbf{now}}$ / $\textbf{capacity}_{\textbf{before}}$ for the a fixed-size chip. Use doubling period to be two years.

b) **Multi-level Computer**
We have a 5 level computer. Each instruction at level n (n=2, 3, 4 or 5) is translated into n+1 instructions at level n-1. For example, one level 3 instruction is translated into 3+1=4 level 2 instructions. Assume there are two types of instructions at level 1 and each type has a 50% share in a typical program. It takes 3 clock cycles (think microinstructions) for an instruction of type A and 5 clock cycles for type B to execute at level 1. What is the program completion time for a 10000-instruction program at level 5 if the clock frequency is 200 MHz?

c) **Pipelines**
Find the processor bandwidth and instruction latency for a system with a 6-stage pipeline and 5 ns cycle time. Assume that no significant stalling occurs due to conditional branching in this

system. Compare this system to another with the same instruction latency but without pipelining by calculating the ratio of the processor bandwidths ($\mathbf{BW_{pipeline}}$ / $\mathbf{BW_{none}}$).

d) **Cache**
In a CPU - single cache - memory system, initially the hit ratio is measured to be **h** and the mean access time is $\mathbf{T_1}$. It is known that memory access time is 8 times that of the cache. By increasing the size of the cache of the same system (same cache and memory access times), the hit ratio is measured to be 25% higher compared to its original value of **h** and the mean access time of the combined system becomes $\mathbf{T_2}$. If $\mathbf{T_1/T_2}$ = 2, find the original hit ratio **h**.

e) **Hamming Code**
The following message has been received as a Hamming code: 10110100 11110000 10101 (spaces are for ease of reading). How many message bits and check bits does this code contain? Recover the original message (i.e. data bits - the message that does not contain check bits) assuming that no more than one bit may have been corrupted during transmission. Use the convention utilized in class.

f) **Endianness and Number Representations**
A number with value $(2584)_{10}$ in pure binary representation is currently stored in memory using a 16-bit word little-endian format with 8-bit cells. What is the new value if this number is mistakenly read using a big endian format again with 16-bit words, 8-bit cells and interpreted as a two's complement number?

g) **Bandwidth and Error Correction**
Two computer systems experience the same fixed bandwidth for their internet connections. System A sends a message with 1-byte words with no error correction and the application on system B sends the same message with 1-byte words (m=8), each encoded with a Hamming code (length of n = m + r). Assume the bits are streamed back-to-back in both systems. Find the ratio of the effective bandwidths of two applications (BW_A/BW_B). The effective bandwidth is defined as the bits of actual data (without check bits) transferred per unit time.

h) **RAID**
Compare the unit cost of storage for two systems by finding the ratio of their costs per byte ($\mathbf{C_1/C_2}$) of usable capacity. System 1 has 10 disks in RAID 1+0 (2 groups) configuration and system 2 has 10 disks in RAID 6 configuration. Both systems utilize identical disks with equal capacity.

Q2 (20%) In this part you will design a 2-bit multiplier. The circuit will multiply two pure binary numbers **A**=$\mathbf{a_1\ a_0}$ and **B**=$\mathbf{b_1\ b_0}$ to produce **M**=$\mathbf{m_3m_2m_1m_0}$, where '**a**'s, '**b**'s and '**m**'s represent bits and their subscripts represent their significance. E.g. **A**=10, **B**=11=> **M**=0110; 2x3=6.

a) Give the truth table for M (all 4 bits), obtain the Boolean expressions for each function and simplify them using algebraic manipulation. Implement using only AND and OR gates (number of inputs as required) and inverters. Show all work.
b) Implement $\mathbf{m_1}$ using an 8-to-1 multiplexer and inverters only.

Q3 (20%) Consider the IJVM assembly program segment below:

```
        …
        BIPUSH 0
        ISTORE i
        BIPUSH 0
        ISTORE j
L1:     BIPUSH 6
        ILOAD j
        ISUB
        IFLT L2
        ILOAD i
        BIPUSH 3
        IADD
        DUP
        IADD
        ISTORE i
        IINC j 2
        GOTO L1
L2:     …
```

a) Give a Java or Python code segment that is equivalent to the IJVM code above. Assume i and j are variable names and represent the same variables in Java and IJVM.

b) Give the list of bytes representing this program fragment in the method area (in hex) assuming that the program segment shown starts at address 100H. Write the corresponding bytes across from each line for the program above (see the class handout: ex4 opcodes and branch addresses). Calculate all relative branch addresses – show your calculations.

c) Provide a hand-written trace of the program by showing values of the variables and the address of the corresponding instruction which caused the change (value shown after the instruction is executed). That is, write a new line in the table below only when the value of a variable changes.

Address of instruction leading to change (hex)	`i` (hex)	`j` (hex)
100	-	-
102	0	-
106	0	0
117	6	0
…	…	…

d) (4-point bonus) Calculate how long it will take to execute this program if the Mic-l has a 1-MHz clock. List the number of clock cycles needed to execute each IJVM instruction (e.g. IADD takes 4 cycles – including the Main1 dispatch cycle; see the microprogram for other instructions).

Q4 (20%) For this question you will write an IJVM program that calculates the area of a rectangle. Your program will take the length and width values from the user (in hex; unit is feet) and print the object’s area (in hex) followed by the unit abbreviation ‘ ft^2’ on the following line. For example, if the user enters **`1A`** and **`6D`**, the output will be **`00000B12`** and **`ft^2`** on the following line. The program should also display the string '**`SQUARE`**' only if length and width are equal to each other. Once the result is displayed it will go to the beginning of the program and wait for a new set of numbers from the user.

Your program should include a method called **mult** which takes two parameters and returns the product of them. You may use successive addition to perform the multiplication (4*5 can be calculated by adding 4 to an accumulator 5 times). You may copy and use the '**getnum**' method in **add.jas** to get the inputs and the '**print**' method to display the hex output for the area.

Suggested program development steps:

1. Study the examples covered in class before starting to write your program.
2. Store arbitrary values in two variables (e.g. length & width) using BIPUSH and ISTORE.
3. Write the code to multiply these two numbers using successive addition and store the result in a third variable. Verify, and debug if necessary, by running it step by step and checking the changes in the variables after each step.
4. Add the print method to your program (copy-paste from add.jas). Using INVOKVIRTUAL, pass the result of the multiplication to the print method to display the number in the output window. Verify that the correct number is being displayed before moving on to the next step.
5. Move the multiplication code into a method called **mult**. Call the method with two parameters and then pass its return value (the product) to the print method. Verify the result.
6. Now use getnum to get a number from the user. Overwrite one of the variables (length or width) with this value to test the operation of the getnum method. Test the code.
7. Complete the code to calculate and display the area (in main by calling getnum, mult and print).
8. Write code to check if both numbers input by the user are equal to each other. Display the string '**`SQUARE`**' if they are equal.

(This order will help you develop the program incrementally. If you can't get the entire program to work, submit the last working step to get a partial grade)

Upload your jas program on the Moodle site before your specific exam deadline. **Important:** Make sure to include comments explaining key points in your program. Put your name in the internal documentation and check that the file you are uploading is a text file containing the IJVM program with extension .jas (not .ijvm).

FINISH

Write your end time, check your pledge and signature.

SUBMIT your jas and answer sheet in pdf on Moodle