THEORY OF INFORMATION, ARCHITECTURE OF COMPUTERS AND OPERATING SYSTEMS (TIACOS)
Bioinformatics, ESCI
DAC, UPC

Midterm exam **Spring 2023**

- This exam is **open book**, **closed Internet**. You can use your laptop, but do not connect to the Internet until you are told to do so.
- Write your answers in a text file named `<your_name>.<your_id>`
- Time **64 minutes**
- You will have 8 minutes at the end of the test to upload your text file to `aula.esci.upf.edu`

1. (2 points) **Short questions**
   (a) About von Neumann architecture
      1. Why its name?
      2. Explain its parts.
      3. Which part is volatile. Why?
      4. This architecture was based on the work of two engineers of the ENIAC project. Name one.

      1. After the Hungarian mathematician John von Neumann (Neumann János Lajos).
      2. Memory, CPU and Input/Output.
      3. Memory, because stored information will be lost without a constant flow of electricity.
      4. John Presper Eckert and John Mauchly,

   (b) Inside CPU.
      1. What is for the Instruction Address Register (also known as Program Counter or Instruction Pointer)?
      2. What is for the Instruction Register?
      3. What CPU does during the fetch phase?
      4. What CPU does during the decode phase?

      1. Contains the address of the current instruction.
      2. The IR contains the current instruction
      3. It loads (copies) the instruction from memory to the IR.
      4. It figures out what the instruction is.

2. (2 points) **Integer representation**

   Write the unsigned (base 10) 8-bits number **170** as binary (base 2), and hexadecimal (base 16). Last, interpret its binary form as a signed 2's complement (base 10) integer. Show all calculation steps.

   1. Binary (base 2):

$$\begin{aligned} 170 &= 2 \cdot 85 + \mathbf{0} \\ 85 &= 2 \cdot 42 + \mathbf{1} \\ 42 &= 2 \cdot 21 + \mathbf{0} \\ 21 &= 2 \cdot 10 + \mathbf{1} \\ 10 &= 2 \cdot 5 + \mathbf{0} \\ 5 &= 2 \cdot 2 + \mathbf{1} \\ 2 &= 2 \cdot 1 + \mathbf{0} \\ 1 &= 2 \cdot 0 + \mathbf{1} \end{aligned}$$

$$y = \sum_{i=0}^{7} y_i \cdot 2^i \Rightarrow \mathbf{170_{10} = 2^7 + 2^5 + 2^3 + 2^1 = 10101010_2}$$

   2. Hexadecimal (base 16): $0b\overbrace{1010}^{0xA}\overbrace{1010}^{0xA} = \mathbf{0xAA}$
   3. Signed 2's complement (base 10): $y = -y_7 \cdot 2^7 \sum_{i=0}^{6} y_i \cdot 2^i = (-1) \cdot 2^7 + 2^5 + 2^3 + 2^1 = \mathbf{-86}$

3. (2 points) **Floating point representation**

   Consider three floating-point numbers A, B and C as per IEEE-754 single-precision floating point format. The 32-bit content stored in these variables (in hexadecimal form) are as follows:

   A = $0xC1400000$

   B = $0x42100000$

   C = $0x41400000$

   Which one of the following is **false**?

   1. $A + C = 0$
   2. $C = A + B$
   3. $B = 3C$
   4. $(B - C) > 0$

   Show all calculations steps, at least, for one variable.

   Answer: Option 2 : C = A + B

   Why? The given data. $A = -12, B = 36, C = 12$. Let's see the steps.

   Decimal value $=(-1)^s * 1.M * 2^{(BaseExponent-Bias)}$. Bias value in IEEE single-precision format is 127.

$$A = 0xC1400000 = \overbrace{1100}^{0xC}\overbrace{0001}^{0x1}\overbrace{0100}^{} \; 0x4\overbrace{0000}^{0x0}\overbrace{0000}^{0x0}\overbrace{0000}^{0x0}\overbrace{0000}^{0x0}\overbrace{0000}^{0x0}$$

$$A = \overset{sign}{\overbrace{1}}\ \overset{exponent}{\overbrace{10000010}}\ \overset{mantissa}{\overbrace{10000000000000000000000}}$$

$$Sign: negative, e = \overset{exponent}{\overbrace{130}} - \overset{E_{max}}{\overbrace{127}} = 3, frac = \overset{normalized}{\overbrace{1}} . \overset{mantissa}{\overbrace{1}}$$

$$A = (-1)^1 * 1.1 * 2^{130-127} = -1.1 * 2^3 = -1100 = (-12)_{10}$$

$$B = 0x42100000 = 01000010000100000000000000000000$$

$$B = \overset{sign}{\overbrace{0}}\ \overset{exponent}{\overbrace{10000100}}\ \overset{mantissa}{\overbrace{00100000000000000000000}}$$

$$Sign: positive, e = \overset{exponent}{\overbrace{132}} - \overset{E_{max}}{\overbrace{127}} = 5, frac = \overset{normalized}{\overbrace{1}} . \overset{mantissa}{\overbrace{001}}$$

$$B = (-1)^0 * 1.001 * 2^{132-127} = +1.001 * 2^5 = +100100 = (+36)_{10}$$

$$C = 0x41400000 = 01000001010000000000000000000000$$

$$C = \overset{sign}{\overbrace{0}}\ \overset{exponent}{\overbrace{10000010}}\ \overset{mantissa}{\overbrace{10000000000000000000000}}$$

$$Sign: positive, e = \overset{exponent}{\overbrace{130}} - \overset{E_{max}}{\overbrace{127}} = 3, frac = \overset{normalized}{\overbrace{1}} . \overset{mantissa}{\overbrace{1}}$$

$$C = (-1)^0 * 1.1 * 2^{130-127} = +1.1 * 2^3 = +1100 = (+12)_{10}$$

4. (2 points) **Shell**

   In a Linux system, the password file (`/etc/passwd`) has contents similar to this:

```
root:x:0:0:root:/root:/bin/bash
sync:x:4:65534:sync:/bin:/bin/sync
john:x:1000:1000:john,,,:/home/john:/bin/bash
alice:x:1001:1000:Alice,,,:/home/alice:/bin/zsh
bob:x:1002:1000:Bob:/home/bob:/bin/tcsh
```

   Type a command line that prints only the username of all the system's users.

```
awk -F: '{print $1}' /etc/passwd
```

5. (2 points) **Pyhton**

   Program in Python a function called `count_bits(num)` that counts the number of set bits (bits set to 1) in a given argument `num`. Examples: `count_bits(7)` = 3, `count_bits(8)` = 1 . No restrictions at all.

```
def count_bits(num):
    count = 0
    while num:
        count += num & 1
        num >>= 1
    return count
```