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

Midterm exam **Spring 2024**

- This exam is **open book**, **open Internet**. You can use your laptop, and access to the Internet, but you cannot chat with other natural or artificial intelligences.
- 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. (3 points) **Short questions**. Justify your answers.
   (a) What is the primary purpose of a cache in a computer processor?
      I. To enhance the disk storage capacity.
      II. To provide backup storage for the main memory.
      III. To reduce the average time to access data from the main memory.
      IV. To increase the processing power of the GPU.

      III.

   (b) Which of the following phases is NOT directly involved in accessing data from memory?
      I. Fetch (FE)
      II. Decode (DE)
      III. Execute (EX)
      IV. Store (ST)

      II, the instruction is broken down into its component parts (opcode, operands) or III, performs the operation, may use data from registers but not directly from memory.

   (c) Scenario: Imagine a processor is about to execute an instruction at memory address 100. This instruction is a JUMP instruction that tells the processor to jump to address 200. Question: After the JUMP instruction at address 100 is fetched and decoded, what happens to the Program Counter (PC) and the Instruction Register (IR)?
      I. The PC remains at 100, and the IR is loaded with the instruction at address 200.
      II. The PC is updated to 200, and the IR is cleared.
      III. The PC is updated to 200, and the IR remains loaded with the JUMP instruction.
      IV. The PC is not modified, and the IR is loaded with the next instruction in sequence (at address 101).

III. The processor fetches the instruction at address 100 (the JUMP instruction). During the decode phase, the processor understands it's a JUMP instruction and needs to change the execution flow. The Program Counter (PC) is updated with the target address of the jump (200). This tells the processor where to fetch the next instruction from. The Instruction Register (IR) typically holds the currently decoded instruction. Since the JUMP instruction has served its purpose, the IR might be overwritten with the next instruction fetched from the new address (200) in the next cycle, but it remains loaded with the JUMP instruction for the current cycle. Therefore, after processing the JUMP instruction, the PC points to the new location (200), and the IR still holds the JUMP instruction that caused the jump.

2. (2 points) **Filling the gaps**

Write the following 8-bits numbers as unsigned (base 10), binary (base 2), hexadecimal (base 16), and signed (base 10) integers. Show all calculation steps.

Unsigned (base 10)	Binary (base 2)	Hexadecimal (base 16)	Signed (base 10)
15	**00001111**	$0F$	**15**
149	**10010101**	95	$\mathbf{-107}$
183	**10110111**	$B7$	$\mathbf{-73}$
73	01001001	**49**	73

- Unsigned: ordinary binary representation

$$y = \sum_{i=0}^{8-1} y_i \cdot base^i$$

  - Range: from 0 to $2^8 - 1$
- Signed: two's complement

$$y = -y_{8-1} \cdot 2^{8-1} \sum_{i=0}^{8-2} y_i \cdot 2^i$$

  - Range: from -2^{8-1} to $2^{8-1} - 1$

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

Suposse we are using the standard IEEE 754, 32b, normalized. Which is the real number represented by the number $0xc2004000$? Show all calculation steps.

$$0xc2004000 = 0b \overbrace{1}^{sign} \overbrace{10000100}^{exponent} \overbrace{00000000100000000000000}^{mantissa}$$

$$Sign : negative, e = \overbrace{132}^{exponent} - \overbrace{127}^{E_{max}} = 5, frac = \overbrace{1}^{normalized} . \overbrace{000000001}^{mantissa}$$

$$real : -1.000000001 * 2^5 = -100000.0001_2 = -(2^5 + 2^{-4}) = -\mathbf{32.0625}_{10}$$

4. (1 point) **Shell**

   In a Linux system, the password file (`/etc/passwd`) contains one line (row) for each local user. The 6th column is the user's home directory. This file 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 lists directory contents of each user's home directory.

```
cat /etc/passwd | awk -F: '{print $6}' | xargs -I{} ls {}
```

5. (2 points) **Pyhton**

   Program in Python, a function named `floatAbsVal(f)` which returns bit-level equivalent of absolute value of `f` for floating point argument `f`. Both the argument and result are passed as unsigned int's, but they are to be interpreted as the bit-level representations of single-precision floating point values. When argument is `NaN`, return argument `f`. Example:

```
>>> floatAbsVal(0xc2004000)
1107312640
```

```
def floatAbsVal(f):
      mask = 1 << 31
      abs = f & ~mask
      if (abs > 0x7F800000):
            return f
      return abs
```